(4)

SRL-0047-TM

AR-005-968

AUSTRALIA

DEPARTMENT OF DEFENCE

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION

SURVEILLANCE RESEARCH LABORATORY

SALISBURY, SOUTH AUSTRALIA

TECHNICAL MEMORANDUM

SRL-0047-TM

STATUS REPORT ON IMAGE INFORMATION SYSTEMS AND IMAGE DATA BASE TECHNOLOGY

M.J. FIEBIG

APPROVED FOR PUBLIC RELEASE

COPY No.

DECEMBER 1989

CONDITIONS OF RELEASE AND DISPOSAL

AR-005-968

DEPARTMENT OF DEFENCE

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION

SURVEILLANCE RESEARCH LABORATORY

SALISBURY SOUTH AUSTRALIA

TECHNICAL MEMORANDUM

SRL-0047-TM

STATUS REPORT ON IMAGE INFORMATION SYTEMS AND IMAGE DATABASE TECHNOLOGY

M.J. FIEBIG

SUMMARY

The contents of this report form part of a study to determine guideline's for the establishment of databases containing digital imagery and ancillary information. It begins with an introduction to the basic concepts of databases. This provides the background to the terminology commonly used in this report and other related literature. A review of current commercially available database systems is made, drawing principally on a market study conducted in mid 1989. Of specific interest is the extent to which image databases are being utilised and the applications which use digital imagery (or pictures) in database management systems (DBMS). Finally, there is an overview of image compression techniques, considering the current trends and future directions of the field.

POSTAL ADDRESS: Director, Surveillance Research Laboratory,
PO Box 1650, Salisbury, South Australia 5108

CONTENTS

PREFACE

The contents of this report form part of a study to determine guidelines for the establishment of databases containing digital imagery and ancillary information. The timeliness of this study is exemplified by the series of good related articles that have appeared since its completion.

Part I provides an introduction to the basic concepts of databases and how relational database management system(s) (RDBMS) vary from non-relational structures. Chapter 1 covers the database architectures. The database models (hierarchical, network, or relational) are the logical structures which the database provides at the user interface. The database uses the file organisations of the operating system or creates the logical structure to provide the data access methods required. Chapter 2 provides an overview of SQL, the 'Structured Query Language', used for data definition, manipulation and control within a relational database. The background given in Part 1 is there to explain the terminology and specific meaning of the questions which form the market survey of commercially available RDBMS in Part II.

Part II looks at current and future database systems, drawing principally on a market study conducted in mid 1989. The results of the survey form Chapter 3. It is organised as follows:
Section 1 gives a brief introduction to give the background of the survey.
Section 2 contains a summary of the RDBMS considered, including an indication of their penetration into the Australian market.
Section 3 details the operating systems for which the RDBMS are available and any hardware or software restrictions that may exist.
Section 4 covers questions relating to relational features in a database.
Section 5 covers underlying database support such as size limitations, support for distributed databases, compression, encryption, security, and datatypes.
Section 6 examines retrieval facilities available, and
Section 7 considers database access currently available from the Apple Macintosh.

Chapter 4, Image Database Management Systems gives an overview of the extent to which image databases are being utilised and some applications for which they have been chosen. Also included are indications of the direction expected over the next couple of years according to scientific publications.

Any database intending to store large amounts of digital imagery must utilise space efficiently. Only recently have commercial databases begun realising the benefits of compression techniques, however most offer little scope (compression of trailing blanks can hardly be called significant compression) and data compression choices remain user defined and implemented. It is important that consideration be given to all possibilities for efficient data storage. Part III gives an overview of image compression techniques, considering the current trends and future directions of the field.

At the end of each part are the references and bibliography relevant for that part

Appendix 1 contains Codd's 12 rules for relational databases, with a few comments relating these rules.
Appendix 2 contains the CCITT standards for facsimile encoding.

Merrilyn.J. Fiebig
Optoelectronics Division
Sept 1989

PART I: BASIC CONCEPTS

"If the auto industry had done what the computer industry has done in the last 30 years a Rolls-Royce would cost $2.50 and get 2,000,000 miles per gallon"

Computerworld

THIS PAGE INTENTIONALLY LEFT BLANK

1. DATABASE ARCHITECTURES

1.1 Logical Data Structures/Data Models

Hierarchical

Data in the hierarchical model is organised in simple tree structures, which an hierarchical database orders into a collection of trees (a forest). As a result the entire database can be considered as a single tree, in which every data item or field is "owned" by a higher ranking item and access to it must be routed through that hierarchy. Any given record takes on its full meaning only when seen in the context of the hierarchy. Hierarchical systems (and inverted systems) were not originally constructed from predefined data models, rather models for these systems were abstracted from implemented systems. The data model of an hierarchical database was largely taken from IMS (Information Management System, IBM). The hierarchical data model is seen as of historical interest only, and it seems unlikely that it will form the basis of future database management systems (DBMS).

Network

A network structure may be regarded as an extended hierarchical structure. It extends the hierarchical constraint from each member (child) having one owner (parent), to each member having any number of owners (including zero). In this model, information is connected via links to form a directed graph structure. Records are grouped into sets, each of which consists of one owner record and zero or more member records. This increases the one-to-one, one-to-many relationship of the hierarchical model to allow many-to-one relationships, however the inability to easily accommodate the many-to-many relationship is still a major deficiency. An example of a network database is IDMS (Integrated Database Management System, Cullinet).

Both network and hierarchical models create subordinate relationships between owner and member data, using pointers to establish and maintain the relationships. Network models are more flexible than hierarchical, and in general more symmetrical, but hierarchical models are simpler. The data manipulation languages (DML) for hierarchical and network databases are procedural, and as such much of the internal data organisation must be visible to the user. The user must be familiar with the lower level of data representation, both how it is declared and how it is stored, in order to develop efficient applications. Consequently, operations are more complicated than on a relational database.

Inverted List

An "Inverted List Data Model" does not exist as a classical model, however it can be considered as a major database structure. An inverted list is similar to a relational database. It contains a collection of files or tables, divided into rows (records) and columns (fields) as in the relational case. However the rows of an inverted list table are considered to be *ordered* in some physical sequence, indexes are not "transparent to the user", and an ordering may also be defined for the total database. The addition and deletion of records requires index maintenance. Although this is handled by the DBMS it can require considerable overhead. No general integrity rules are included in the inverted list model.

Networks and hierarchies are structured around the use of pointers to establish the relationships between the data; inverted list structures are characterised by establishing relationships outside the database. No internal pointers are embedded within the database records. Data restructuring has less impact and new fields can be added without requiring the reprogramming of existing applications. An example of an inverted list database is Adabas (Software AG).

Relational

The relational model has its roots in mathematical set theory and was formally introduced by Dr E.F. Codd in 1970 [1.1]. The solid theoretical foundation is reflected in the model's inherent simplicity. The relational model consists of three components: data structure, data manipulation and data integrity. Acceptance of the relational model accelerated slowly. Following Codd's guidelines, structured query languages evolved and relational databases became more popular. In 1985, to inject some order into the rapidly increasing literature on relational databases, Codd laid down 12 principles [1.2],[1.3]. At least six of these must be satisfied before a database can be considered relational. Codd's 12 relational rules are provided in Appendix 1.

The data is perceived by the user as relational tables and the operations performed generate new tables from combinations of old. Each relational table consists of uniquely named columns (attributes) containing single-valued entries of the same datatype (domain) and an arbitrary number of unique rows (tuples). The special properties of relational tables also includes the stipulation that the order of the columns and rows is immaterial. That is, the sequence in which the columns or rows are stored does not affect the meaning of the data.

The data manipulation language (DML) is based on the applied predicate calculus and designed to operate on relationships. It includes the assignment of relations (insert, update, delete) and manipulation using relational operators (select, project, product, join, union, intersection, difference, division). The database management system (DBMS) and all its utilities use a common data dictionary. This dictionary is a database in its own right. It is managed by the DML and stores predefined queries, definitions of relations, attributes and access permissions.

The relational model includes various integrity rules and constraints, which it implies should be addressed as part of the database implementation, not as part of the application implementation. Integrity rules ensure the data is complete, but not redundant. For example, the entity integrity rule does not allow nulls (values unknown or not supplied), in the primary key[1]. The referential integrity rule is the second very significant rule to receive recent support in commercial databases. Referential integrity ensures that if a foreign key[2] in one table is a primary key in another table then every value matches a value in the relational table in which that foreign key is a part, or is null.

The main attraction of the relational model is its mathematical clarity, which facilitates the formulation of nonprocedural, high level queries and thus separates the user from the internal organisation of the data. The relational data model represents the dominant trend in the market today. Many older systems developed before relational systems became dominant have undergone retrofits to provide some kind of relational support. Almost all recent database development has been relational and all the particular databases considered in this report are relational.

---

[1] The primary key is the column or set of columns uniquely identifying a set of rows.

[2] A foreign key is the column or set of columns in one table that is not a key in that table but is a key elsewhere (e.g. in another table). Used for relating data in multiple tables using joins.

1.2 File Organisations

The file organisation of the data structure is a logical structure which corresponds directly to the access method being used. If an operating system does not have the capabilities required by the DBMS, the DBMS must perform those file access operations not possible with the operating system alone.

Indexed files can be randomly ordered or sequentially ordered. In sequential access, the program reads (or writes) the file from the beginning, accessing all records in succession. With random (also referred to as 'direct') access the program can reach a record according to its rank, without accessing previous records first. Generally sequential files can have variable length records which can only be opened in Read *or* Write mode, while random files (restricted to fixed length records) can be opened in Read *and* Write mode. However the Unix operating system is conceptually different and does not have these restrictions. In Unix the system considers any file to be a simple, undifferentiated sequence of bytes. Thus anything that can be represented as a stream of bytes - text, executable programs, bit-maps, mail messages, even output devices - are files. It is up to the user or an application program to impose any addition structure required.

An indexed-sequential file provides fast retrieval, however it is slow to update. Sequential files are not used for permanent storage by a DBMS, because they cannot be updated and do not provide direct access. The Unix file structures may be difficult to use with application programs but its differences from the classical description mean that Unix is the 'perfect' operating system for use with a DBMS [1.4].

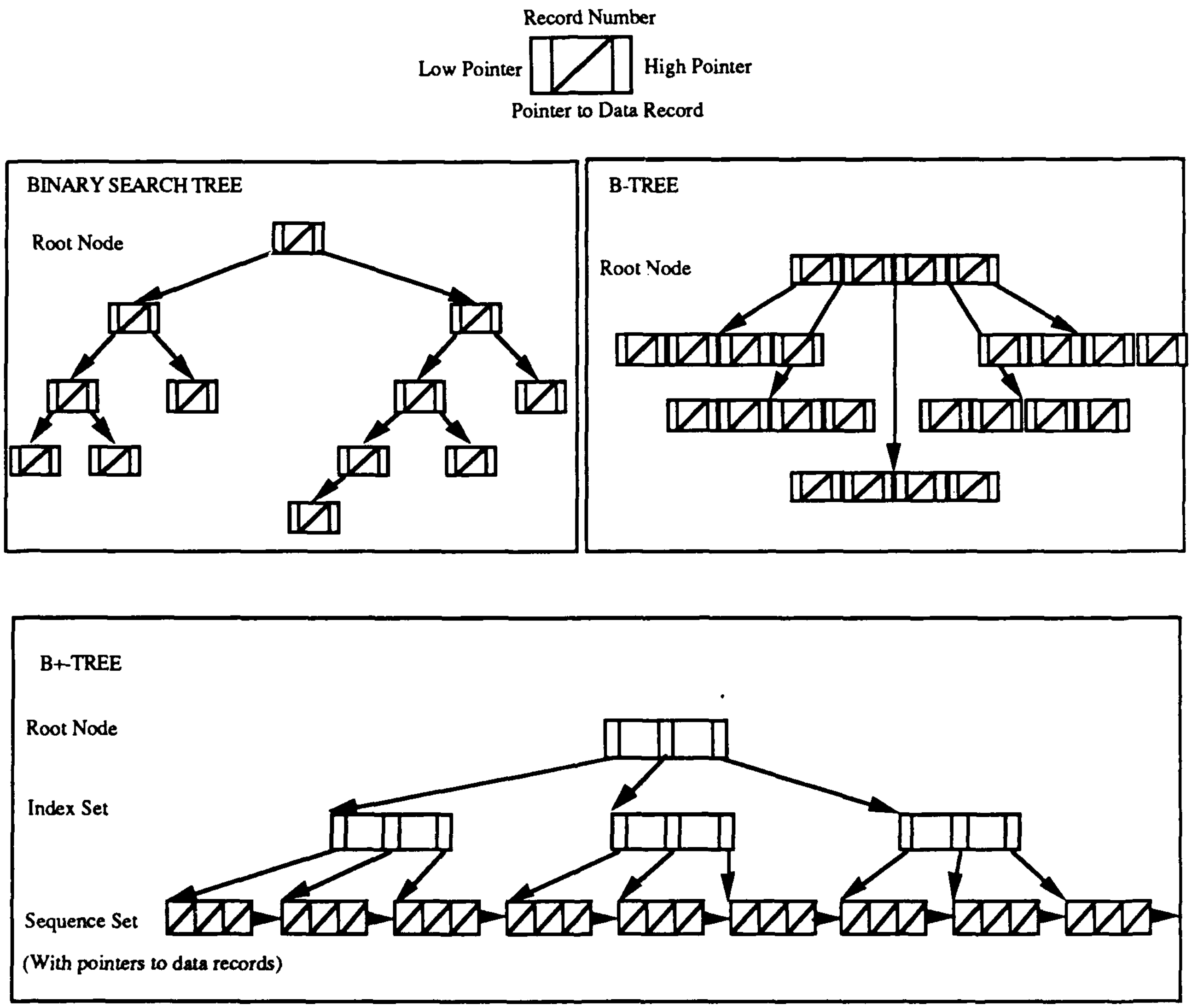
List, ring, tree, next and network file organisations use **pointer structures**. The list file is the most elementary pointer structure. All the records in list files are related by pointers. The list is processed in logical sequence rather than physical sequence. Ring or chain files are extensions of the list structure. If the last record in the list has a pointer back to the top of the list, creating a continuous loop, the file is a ring or chain file. Forward and backward pointers may provide access in either direction.

Tree files [1.5] are hierarchical structures which provide access to data through the use of pointers. **Binary trees** (see figure 1.) are not used because they require high overheads to provide efficient access and result in long retrieval times. **Multiway trees** are a generalisation of binary trees. Instead of containing one record and pointers, each node contains R records and R+1 pointers. **B-trees** are balanced multiway trees. Although there is no single structure that is optimal for all applications, the B-tree (or variation) is generally accepted as the optimal performer. In addition to being multiway, B-trees have efficient self-balancing operations and so do not have the retrieval or maintenance problems of binary trees. B-trees provide logarithmic search time, automatic reorganization, and good space utilization [1.6]. Variations of the B-tree include B*-trees and Knuth's variation of the B-tree which has become known as the B+-tree.

In a **B*-trees** all terminal nodes appear at the same level, that is, they they are all the same distance from the root. B*-trees have a higher branching factor on average than B-trees. As a result, searching a B*-tree is faster on average than searching a B-tree. Thus B*-trees are better when searching is the most common tree operation, while B-trees are better for applications where insertions or deletions are more common than searches.

The **B+-tree** has two distinct parts. Records are kept at the terminal nodes. The non-terminal nodes form the index part of the tree and contain only key values and tree pointers. These nodes have a different structure from the terminal nodes which do not need tree pointer fields. All searches terminate at the lowest level of the tree. However, because the index levels hold only keys rather than complete index records, searching time is comparable with the B-tree holding the same number of records. The tree remains balanced and the record keys are arranged in sequential order. Getting to the next record in sequence in a B+-tree requires at most one node read. Therefore, B+-trees are good in applications in which both direct and sequential processing are required.

Figure 1: B-Tree data access methods evolved from the binary search method.



Object orientated databases require structures that can efficiently handle 2-D imagery and spatial objects of 2-D or higher. R-trees [1.7] are B-trees extended to 2-D. They are similar to quadtrees but more flexible.

Net files use a combination of tree and ring structures. The combination of pointing and chaining results in high overheads.

1.3 Data Access at the Physical Level

Database access can occur through the use of **key indexes**. Indexes speed up retrieval but slow down updates. Index maintenance is required whenever an entry is altered or deleted. It can be *automatic* or *manual* (static or dynamic). An automatic index is updated by the DBMS each time keys are added, deleted or altered. A manual index implies that it is updated when the program requests it. Most DBMSs can maintain primary and secondary indexes on each file. The **primary index** of a file is the index built on its primary key (see footnote 1 earlier). The **secondary index** of a file is an index built on a field other than the primary key field. A file with an index on every field is said to be fully inverted.

The **B-tree** structures described earlier can be used for both storage and indexing of the database. Indexes are often tree structured because balanced multiway trees offer efficient search operations and easy sequential accessing. They are an alternative to a nonsequential index and provide one of the fastest sorting techniques available.

Hash coding may be used for accessing the data or for indexing. The key value (or index), unique or not unique, is converted into a record number using a mathematical transform and an algorithm. A very common class of hash function is "division/ remainder". The hash address is the remainder after some field (the hash field) of that record (not necessarily but usually the primary key) is divided by a prime number. The DBMS computes the hash address and stores the record at that position. To retrieve the record, the DBMS must recompute the hash field value to fetch the record from the computed position. Thus while a given stored field can have any number of indexes, it can only have one direct hash structure.

Hash coding was invented to solve the performance problems of indexing. It decreases disk accesses required to find and update a record. It is particularly effective if it is updating large files, where it can be used to access a hash-coded index file first and then the data file.

Bit inverted files are designed to quickly eliminate records that do not meet the search criteria, leaving only a small number of records that must be examined singularly. A good example of a bit-inverted file is one created on the date field. A bit vector is created for each record, so that

0 => every date > 1 day old
1 => yesterday
2 => today
3 => future dates.

These bit vectors are grouped in blocks to form an index, which can be scanned block by block, bit vector by bit vector. All that is required is a binary AND instruction to extract the bit field from its elemental word, then a comparison with the predefined value. These bit operations are among the fastest on any computer.

Updates are fast and a bit-vector index can be used in conjunction with the other file structures. In his book 'Advanced Database Techniques' [1.4], Martin laments that the only disadvantage is that most DBMS do not use this structure, because DBMS are optimised for business applications that do not have unpredictable queries on many simultaneous fields. So it is left up to Scientific and Engineering application users to program it for themselves. Of the databases surveyed only StarBase supports bit-maps. However it is interesting to note that StarBase also supports BLOBS (Basic Large Objects), implying that diversification of DBMS may lead to greater support in the future.

2. SQL - The Standard Query Language?

Between the Idea
And the Reality,
Between the Motion
And the Act,
Falls the Shadow

George Elliot 1819-1880

Dr E.F. Codd introduced the requirements of a 'structured query language' with the concept of a relational database in 1970 [1.1].

" the adoption of a relational model of data ... permits the development of a universal sublanguage based on an applied predicate calculus. "

E.F.Codd 1970

This created for developers the possibility of a universal sublanguage without explicitly giving the mechanics. Intuitively this would appear to reduce the amount of ad hoc work normally required in language development and lead to an elegantly simple but powerful language.

While ORACLE used SQL from its start (1979), it was not until IBM produced the SQL based DB2 in 1982 that the American National Standards Institute (ANSI) looked at making it a standard. In 1986, after several years of debate, ANSI finally approved definition of a base SQL standard [1.8]. SQL's strongest competition was QUEL from Relational Technology's INGRES. However recently, even INGRES has adopted SQL-based languages. IBM's dominant position in the market place has meant that SQL is the standard and all other query languages, regardless of their intellectual appeal, will slowly fade from prominence.

SQL was originally an acronym, standing for 'Structured Query Language', but SQL is more than a query language. SQL includes a data definition language (DDL), a data manipulation language (DML), and a data control language (DCL). SQL, the data definition language, is the database programming language. This portion of SQL is used by database administrators (DBA). SQL, the interactive query language, provides the facility for retrieval as "query" suggests, but also UPDATE, INSERT, DELETE and other operations as well. This is the end-user level of SQL. The original version of SQL was intended to provide interactive standalone use for end-users. SQL, the embedding in a host language, is the application programmer level of SQL - and it is this area the standard concentrates almost exclusively on.

Unfortunately no two 'standard' SQL implementations will ever be truly identical. SQL fails to support several basic functions, for example there is no DROP TABLE statement (this presents an inability to delete relational tables from the database). This is only one aspect left as implementation-defined rather than part of the standard [1.9]. A Data Dictionary is essential for efficient use of SQL. Its exact form is a system feature - not a part of SQL. Data Dictionary Information (the system catalog) is not addressed by the current ANSI standard. Thus each standard SQL system will have a different representation for the dictionary. Differences exist in the exact forms of indexes and view support facilities, which may influence data base design. Some vendors support default values - others do not. The current SQL standard does not address these issues - limiting the leverage of a DBA. Thus it was out of necessity that vendors produced many implementation-defined extensions and syntax variations. As a result a simple SQL command syntactically correct on one database will probably fail on another, and even if it does not fail there is no guarantee that the interpretation will be identical.

There are about 30 words in the SQL definition/ query/ manipulation language. Oracle contains more than one hundred reserved words, including system commands, the programmer workbench, the format commands, and the database administrator. This does not include system specific facilities such as screen definition facilities, report and graphic specifications. If these are considered ANSI standard SQL comprises only 10-20% of the total specification.

SQL does not have a rigorous implementation. Although SQL does offer a standard, the way in which the query optimiser is implemented can greatly affect the performance of the database. SQL is relationally complete [1.10], but is far from an ideal relational language. The language is filled with restrictions, ad hoc constructs, and special rules. It fails to realise the full potential of the relational model [1.11].

ANSI is moving to correct some of these issues. Topics under consideration as part of the addendum include referential integrity, enhanced transaction management, specification of certain user-defined rules, enhanced character-handling facilities and support for national character sets [1.8]. However even when the addendum[1] is ratified or the proposed SQL-2 standard is made, both are doomed to instantaneous technological obsolescence. Standard SQL will only contain a subset of available commercial functions. SQL only supports three major datatypes: Character (character string with *specified* length), Exact Numeric (decimal, integer, small integer), and Approximate Numeric (float - real and double precision). Although these datatypes may be extended, there is no current ANSI consideration in the areas of knowledge management and object management. So the storage of images or basic large objects (BLOBs) is likely to remain non-SQL. In fact, there is some opposition to the follow-on standard being developed by ANSI (SQL-2), on the basis it will stifle the natural maturation of SQL [1.13].

So 'Who benefits from a standard SQL?', is a question widely asked. End users will use customised interfaces usually of the fill-in-the-form variety[2]. SQL's low level interface offers flexibility, but complex commands quickly become difficult to decipher, especially for the novice user. Application programmers will use a fourth generation language (4GL[3]), because its software development offers greater leverage. (And there is no 4GL standard in sight because IBM does not have a recognised 4GL.) So perhaps it is 4GL vendors who benefit from a standard? All 4GLs must read and write information into a dictionary - for which there is no standard. If anything, when a new ANSI SQL is finally bought in, 4GL vendors will probably be inconvenienced by the retrofit needed.

And yet, in spite of all its short-comings, SQL is the standard, vendors are striving to support it, and customers are demanding its support. Perhaps some hope lies in this. User interest has created a new standards group called "Open SQL" [1.15]. Open SQL is advocating an "X.400 approach" to database interoperability and access, and is looking to provide a common application programming interface by early next year.

---

[1] Addendum 1, X3H2 to the present SQL standard (ANSI X3.135-1986) was expected in 1988 but still has not been ratified.

[2] For example, it is specifically stated that tabular form was chosen for PICQUERY [1.14] because of the difficulty end-users have using a non-procedural SQL-like query language.

[3] 4GLs are database manipulation languages such PowerHouse (COGNOS); 3GLs are the programming languages Pascal, FORTRAN, C etc.

THIS PAGE INTENTIONALLY LEFT BLANK

PART I: BASIC CONCEPTS

References and Bibliography

[1.1] E.F. Codd, A relational model of data for large shared data banks, *Communications of the ACM*, Vol 13, No 6, pp 377-387, June 1970.

[1.2] E.F. Codd, Is your DBMS really relational?, *Computerworld,* October 14, 1985.

[1.3] E.F. Codd, Does your DBMS run by the rules?, *Computerworld,* October 21, 1985.

[1.4] D Martin, *Advanced Database Techniques*, MIT Press, pp 193-194, 1986.

[1.5] P.D. Smith and G.M. Barnes, *Files and Databases: An Introduction*, Addison-Wesley Publishing Co., pp 152-181, 1987.

[1.6] R. Bayer and E. McCreight, Organization and Maintenance of Large Ordered Indexes, *Acta Informatica*, Vol. 1, No. 3, pp 173-189, 1972.

[1.7] N. Roussopoulos, C. Faloutsos, T. Sellis, An Efficient Pictorial Database System for PSQL, *IEEE Transactions on Software Engineering*, Vol. 14, No. 5, pp 639-650, May 1988.

[1.8] American National Standards Institute, *American National Standard for information systems - database language - SQL. ANSI X3.135-1986*, New York: American National Standards Institute, Inc., 1986.

[1.9] C J Date, A Critique of SQL, *SIGMOD Record*, Jan 1985.

[1.10] C.J. Date, *An Introduction to Database Systems*, Vol 1, 4th Edition. Menlo Park, Calif.: Addison Wesley, 1986.

[1.11] C J Date, *A Guide to the SQL standard*, Appendix E, Addison-Wesley Pub. Co., Reading MA, p 155-199, 1987.

[1.12] C.J. Hursch and J.L. Hursch, *SQL - The Structured Query Language*, TAB Books Inc, Blue Ridge Summit, PA, 1988.

[1.13] M Stonebraker, Future Trends in Data Base Systems, *1988 IEEE Fourth Conference on Data Engineering*, p 222-231, Feb 1988.

[1.14] T. Joseph and A.F. Cardens, PICQUERY: A High Level Query Language for Pictorial Database Management, *IEEE Transactions on Software Engineering*, Vol 14, No 5, p 631, May 1988.

[1.15] Open SQL Promised Soon, *Computerworld Australia*, p 13, Oct 13, 1989.

THIS PAGE INTENTIONALLY LEFT BLANK

PART II: DATABASE MANAGEMENT SYSTEMS

A database is a collection of information on a well defined subject that is exhaustive, nonredundant and structured.

Daniel Martin, 1986

THIS PAGE INTENTIONALLY LEFT BLANK

3. A MARKET STUDY

3.1 INTRODUCTION

The scope of this study covers relational databases available and supported within Australia, with the view of using a Database Management System for imagery and ancillary information. The relational database management systems considered in this market study are Cognos Powerhouse Starbase RDBMS, Informix SQL, Ingres, Oracle, VAX Rdb/VMS, and Sybase. Replies were received from the companies in the period August to September 1989.

The questions asked of the companies in this study are based on those used in the 1986 Xephon buyers guide [2.1], which compares relational DBMSs available for IBM mainframes. The questions within that guide are more extensive. The answers from the two studies provides an insight into developments that have occurred over the relatively short period of time between them. Relational Database Management Systems have traditionally imposed severe performance overheads at run-time. While they have offered flexibility, vendors have been keenly aware that performance had to be addressed. This aspect has changed since 1986 and almost all the databases considered are part of the new generation of RDBMS. Extensions include multiple transactions per process, true distributed processing, and increased availability allowing backups while in use, dynamic online restructuring and node independence. Changes that reflect the increase in versatility and broader application can be seen in the ability to use DBMS to define relational views on top of non-relational structures, increased integrity, including triggers for integrity rules, and the increase in datatypes supported. Before DEC's Rdb/VMS became available in 1985 there was very little support for data integrity. In 1986, very few databases included structured text while none provided digitised image or digitised audio as a datatype.

The questions were designed to produce an accurate response over our area of interest rather than a comprehensive coverage of all aspects of relational databases. Areas where differences exist are highlighted by the implications of the answers. However ultimately questions pertaining to ease of use or functionality versus benchmarks are best tested under the applied conditions for which the database is required.

The replies to the questionnaires sent to the above companies have been compiled to allow comparison. The answers have been entered as they were given by the company representative(s). Sometimes additional verbal information was obtained to clarify the answer. Each answer reflects the claims of the company and should also be taken in the context of any associated comments.

3.2. PRODUCT DESCRIPTION

The following product descriptions have been received from the respective companies. Any claims made are made by that company.

INFORMIX SQL

Brief Description:

Informix SQL is a relational Database with the ability to create Databases and Tables on-line and then create Forms and Reports from the database schema.

Country of Origin: U.S.A.

Product Developers: Informix Software Inc.

Vendor - Australia: Informix Software (Australia) Pty. Ltd.

Start of Company: 1980

First Product Installation: 1980

Total Sites with Product: >250,000

No. of sites in Australia: 1000 South Australia: 10 (est.)

Associated Packages Supplied: Informix 4GL, Embedded SQL for C, Cobol & ADA.

Significant Recent Developments:

The next release of Informix turbo will support "BLOBS" (Binary Large Objects). Digitised images, graphs or documents can be stored in the database.

Future Plans:

Informix intends to integrate the database with their two office automation products Wingz (Macintosh) and Smartware II (PC-DOS and Xenix). The integration will allow users of Wingz/Smartware to query the database and retrieve the data into spreadsheets or pre-written applications.

Other Areas addressed by Product:

Informix currently has networking software which allow for True Distributed Processing of data. The next release of Informix Turbo will support distributed Databases as well.

INGRES

Brief Description:

INGRES is a fully functional distributed relational database with a 4GL integrated within the development environment. INGRES is designed and developed by Relational Technology and is accepted as being world leader in relational databases. *(This response seems to be more manufacturers hype than a true assessment of the products position.)*

Country of Origin: U.S.A.

Product Developers: Relational Technology International

Vendor - Australia: as above

Start of Company: Australia 1986 U.S.A. 1981

First Product Installation: Australia 1984 U.S.A. 1981

Total Sites with Product: >13,000

No. of sites in Australia: >150 South Australia: 13

Associated Packages Supplied: INGRES Product only.

Significant Recent Developments: REV 6.1 (enclosed information)

Future Plans:
Integrated Case Tools, A.I., User Tools window based. Open SQL, Open Communications.

Other Areas addressed by Product:
Gateways/Interfaces to other relational and non-relational products.

ORACLE

Brief Description:
ORACLE delivers fully relational capabilities and extended features. It offers a full implementation of SQL and runs on a large range of mainframes, minis and micros. ORACLE's distributed architecture lets data and applications reside on multiple computers and still communicate transparently. It is based on advanced architecture that maximizes throughput, facilitates multi-user transactions and protects the data from both unauthorized access and system failure. ORACLE provides a comprehensive set of utilities for configuring and implementing applications. These utilities can be used to load data from external files, backup and recover selected data, move data from one database to another, monitor ORACLE's performance and control disk space utilization.

Country of Origin: U.S.A.

Product Developers: ORACLE CORPORATION

Vendor - Australia: ORACLE SYSTEMS AUSTRALIA

Start of Company: 1976

First Product Installation: ORACLE RDBMS in 1979

Total Sites with Product: > 15,000

No. of sites in Australia: > 2,000 Australia
(approx. 30 local including DEWADL Australian Submarine Corporation, SACON, Telecom)

Associated Packages Supplied: ORACLE*FINANCIALS

Significant Recent Developments:
Press release of the NCUBE support for massively parallel cpu architecture.

Future Plans:
Seamless integration of text, graphics and attribute information across hardware platforms to enable enterprise wide sharing and to maintain the use of current technology.

Other Areas addressed by Product:
To continue to release application packages for manufacturing, spatial images processing, etc.

RAPPORT

Brief Description:

RAPPORT is a relational database management system providing security and recovery facilities for multi-user environments. It has a number of application generation tools including RAPIDE (fourth generation language), RAPIER (screen based applications), RaSQL (structured query language) and interfaces to FORTRAN, Pascal and COBOL.

Country of Origin: UK

Product Developers: Logica (UK) Ltd

The last version of RAPPORT was RAPPORT-5, which was released in October 1985. Without large research and development resources RAPPORT has fallen behind other major RDBMS and now looks "old fashioned". Logica has felt unable to continue supplying RAPPORT, and while it continues to support existing users most are converting to other RDBMSs.

VAX Rdb/VMS

Brief Description:

VAX Rdb/VMS is a full function relational database management system based on Codd's relational model. It is intended for general purpose, multi-user, centralised or distributed applications. It facilitates easy, interactive restructuring of a database and supports full on-line backup, null values and both single and multifile databases. Applications on a given node in a DECnet network can access databases on other nodes in the network. VAX Rdb/VMS in a VAXcluster environment allows concurrent, multiple-processor database access. VAX Rdb/VMS ensures referential integrity of the data using constraints and VALID IF clauses. It supports 'transaction' and 'record locking' to ensure data consistency and integrity during concurrent accesses.

The Relational Management Utility is a comprehensive utility for monitoring the performance and access to Rdb/VMS databases.

Country of Origin: U.S.A.

Product Developers: Digital Equipment Corporation

Vendor - Australia: Digital Equipment Corporation (Australia)

Start of Company: 1957

First Product Installation: November 1984

Total Sites with Product: >9,000

No. of sites in Australia: 552 (inc. NZ) South Australia: 9

Associated Packages Supplied:

VAX Rally (4GL), VAX TEAMDATA (decision support), DECdecision (decision support), VAX Datatrieve (report writer). DEClink (interoperation with IBM databases). VAX SQL Services (a client server model for remotely executing SQL instructions on an Rdb database, eg. from a program on IBM PC or PS/3).

Significant Recent Developments:
Support for multifile database and multi-disk database (June 1988). Support for horizontal partitioning (June 1988). Support for record clustering - groups of records which are frequently accessed together may be stored on the same physical database page to reduce I/O (June 1988).

Future Plans:
Future plans are Digital Confidential.

Other Areas addressed by Product:
The run-time licence for Rdb/VMS is provided at no charge with the VAX/VMS licence for VMS version 5.1 and later.

COGNOS PowerHouse Starbase RDBMS (from here referred to as StarBase)

Brief Description:
Second generation Relational Database developed using advanced relational technologies to provide the true distributed database functionality ANSI and ISO standard SQL compliant, offers full distributed networking, highest levels of data integrity, 3GL support, Digital Systems support.

Country of Origin: CANADA

Product Developers: COGNOS/INTERBASE CORP

Vendor - Australia: COGNOS

Start of Company: 1968 - COGNOS INC. OTTOWA CANADA

First Product Installation: Powerhouse 4GL-1981, Starbase - 1988

Total Sites with Product: 100+

No. of sites in Australia: 10 South Australia: 1

Associated Packages Supplied: PowerHouse, StarGate, StarNet.

Significant Recent Developments: Acceptance major teaching Universities (Australia), U.S.A.F. Major Corporations.

Future Plans:
Port to UNIX, IBM, HP3000, HPPA, DG.

Other Areas addressed by Product:
Distributed Processing, Multi Tier Environments, Integration with multiple vendors RDBMSs, high performance.

SYBASE

Brief Description:

The backbone of the SYBASE system is its advanced client/server architecture which enables application functions to be handled independently from data management functions in both standalone and networked environments. SYBASE's capabilities allow it to accommodate a variety of applications where other relational products may be limited. It has two components: the SQL Server™ and the SQL Toolset™. SQL Server and SQL Toolset offer the following five key capabilities: scalable high volume performance, server enforced integrity, high availability, open distributed DBMS and window-based tools.

Country of Origin: USA

Product Developers: SYBASE INC

Vendor - Australia: SYBASE AUSTRALIA Pty Ltd

Office is currently Sydney based, Melbourne Office due to open 2-3 months time, Canberra within 12 months.

Start of Company: 1984

First Product Installation: June 1986

Total Sites with Product: 450 - 600 customers since 1st shipments in 1987

No. of sites in Australia: 10

Although during SYBASE's first year in Australia, there are no sites in South Australia, current users include QLD Police Dept., Telecom, State Bank NSW, Macquarie and ANZ banks.

Associated Packages Supplied:

None but interfaces are provided for third party products (see SYNERGY documentation).

Significant Recent Developments:

Secure Server - Level B1 & B2

OPEN Server Gateway mechanism to access non-sybase data.

Future Plans:

Sybase is committed to an active program of product development such as an enhanced Central Data Dictionary and an enhanced Report Writer.

Other Areas addressed by Product:

Complete set of development tools including 4GL.

3.3. ENVIRONMENT

3.3.1 Which operating systems or other major subsystems are supported directly?

INFORMIX	PC-DOS, Xenix, Unix (most varieties), VMS. Informix Turbo is only available on Unix.
INGRES	UNIX, VAX/VMS/ULTRIX, PC XT AT IBM.
ORACLE	UNIX System 5, MVS/SP, MVS/XA, VM/CMS, UTS, AEGIS-DOMAIN/IX, Native, A/UX, NOS/VE, CTIX, AOS/VS, DG/UX. UNIX, VMS, ULTRIX, VOS, HP/UX, GCOS, AIX, VME, OSx, SINTRAN, PRIMOS, DYNIX, BS2000, SINIX, OS 3.x, VS, PC-DOS, MS-DOS, XENIX, OS/2. ORACLE is supported upon >90 platforms.
Rdb/VMS	VAX VMS is directly supported. Rdb can interoperate with IBM's DB2, IDMS/R, VSAM and IMS databases through Digital's DEClink software product. VAX DQL Services provides a means by where applications running on other hardware platforms may access RDB/VMS via remote SQL procedure calls.
STARBASE	VAX/VMS, UNIX (HP), OS/2, DG/UX.
SYBASE	UNIX, VMS, VOS, OS/2

3.3.2 Are there any restrictions in using the package with any other software?

INFORMIX	No	
INGRES	None	
ORACLE	None	
Rdb/VMS	No	Further, the interface to Rdb/VMS is published and available publicly. There are over a dozen third party application development tool kits which operate on Rdb.
STARBASE	No	
SYBASE	No	

3.3.3 Are there any software prerequisites?

INFORMIX	No	
INGRES	Yes	Certified operating systems by RTI.
ORACLE	None	
Rdb/VMS	Yes	The only software prerequisite is the VMS operating system.
STARBASE	No	
SYBASE	No	

3.3.4 Are there any hardware prerequisites?

INFORMIX	No	
INGRES	Nil	
ORACLE	Yes	Specific to each part and the required application requirements.
Rdb/VMS	Yes	The only hardware prerequisite is a processor from the VAX range.

STARBASE No (limited only to platforms supported)
SYBASE No

3.4. RELATIONAL DBMS FACILITIES

3.4.1 Does the product permit the use of predefined access path support at the physical level?

The use of predefined access path support at the physical level does not necessarily disqualify a product from the 'relational' club, as long as it is invisible to programs and DBA utilities (except for tuning purposes).

INFORMIX Yes

INGRES Yes. INGRES is not restricted to using pointers - fully relational automatic navigation is used. Eight different storage structures are available.

ORACLE Yes

Rdb/VMS No. The method of access cannot be pre-defined. Indexes may be defined on any field except unstructured datatypes (such as voice). Indexes may be specified as 'duplicates not allowed' (as in primary key) or 'duplicates allowed'. The query optimiser decides on whether sequential or index retrieval will produce the least number of I/O's and chooses the appropriate strategy. If an index is defined, then the optimiser will take this into account when deciding on the most efficient retrieval strategy.

STARBASE Yes

SYBASE Yes

	Primary key index	Secondary indexes	Pointer Arrays	Pointer Chains	Other
INFORMIX	x	x	-	-	-
INGRES	x!	x#	-	-	Hash Access is available as Heap storage.
ORACLE	x	-	-	-	Use of operating system variables.
Rdb/VMS	-	-	-	-	Indexes are provided but there is no physical distinction between primary and secondary.
STARBASE	x	x	-	-	-
SYBASE	x	-	-	-	Any attribute or set of attributes may be indexed. Supported index types are B-trees and B*-trees.

x denotes YES (for all tables)
\- denotes NO (for all tables)
Additional symbols are explained beneath the relevant table.

! 6 of the 8 storage structures are keyed, 4 of these have a primary index.
\# Secondary indexes may be generated dynamically on any secondary key.

3.4.2 Does the product provide a free-standing 'user language' in which the database can be accessed and updated without the use of a normal programming language?

	SQL	SQL superset	Vendor's own syntax	Other
INFORMIX	x	-	-	
INGRES	x	x	x!	Applications by Forms/4GL; Forms-Based and user toolset incorporating Visual Programming techniques.
ORACLE	x	x	-	DB2
Rdb/VMS	x	x	x#	VAX SQL is compatible with both ANSI standard and IBM's DB2 SQL. It includes support for the ANSI SQL module language specification (SQL2).
STARBASE	x	-	x'	PowerHouse
SYBASE	x	x*	-	The APT-SQL 4GL may have TRANSACT-SQL statements embedded.

' GDML

! QUEL

RDO (Relational Database Operator) is the Digital proprietary query language for Rdb/VMS.

* TRANSACT-SQL

3.4.3 Does the user language support access to data in a different DBMS by allowing relational views to be defined on top of a possibly non-relational structure?

INFORMIX No

INGRES Yes INGRES has adopted an 'Intelligent Gateway' strategy and has available for installation now, gateways to: DEC's Rdb, DEC's RMS; and is developing/testing gateways to: DB2, IMS and others.

ORACLE Yes SQL/DS, DB2, IMS, RMS

Rdb/VMS Yes Through Digital's DEClink package these languages may be used to access IBM databases.

STARBASE Yes Rdb, RMS, any other SQL compliant RDBMS.

SYBASE Yes The Open Server product is a toolkit which enables the development of GATEWAYS. These GATEWAYS can be developed to access non-SYBASE and non-relational data.

3.4.4 Does the relational DBMS include an 'embedded' language which can be used within a host programming language? (Cobol, FORTRAN, PL/1, ADA C, Pascal, Basic, other)

INFORMIX Informix has embedded language support for C, COBOL and ADA. These products are not embedded with the database as standard.

INGRES	Yes	There are INGRES Embedded SQL precompilers available to all 7 of these languages.
ORACLE	Yes	Cobol, FORTRAN, PL/1, ADA, C.
Rdb/VMS	Yes	Cobol, FORTRAN, ADA, C, Pascal and Basic are supported for both SQL and RDO. Modular SQL is also supported for all native VAX languages.
STARBASE	Yes	Cobol, FORTRAN, PLS, ADA, C, Pascal and Basic.
SYBASE	Yes	SQL code is "embedded" as stored procedures in the SQL Server which is an active database. These stored procedures are invoked from the 'host' programming language via a library interface.

3.4.5 Which access control facilities are provided:

	DBA granting of access permissions	Authorised Users delegating granting of access permissions	Separate Read & Write Access Control	Access Control by Base Table	" " by View	" " by specified Columns*	" " by specified Rows*	Revoking of Permissions
INFORMIX	x	x	x	x	x	-	-	x
INGRES	x	x!	x	x	x	x	x	x
ORACLE	x	x	x	x	x	x	x	x
Rdb/VMS	x	x	x	x	x	-	-	x
STARBASE	x	x	x	x	x	x	x	x
SYBASE	x	x	x	x	x	x	x	x

DBA	Data Base Administrator
Base Table	Any 'real' table in the database, as opposed to a 'virtual table'.
View	A table that does not physically exist as such in storage, but looks to the user as though it does. A part of a table that does exist in the database. A virtual table.
*	The ability to lock individual columns or rows decreases access time while increasing user accessibility
!	By owners and super-users only.

3.5. UNDERLYING DBMS SUPPORT

3.5.1 What are the practical limits on the size of the database and the level of simultaneous usage?

INFORMIX	Informix Turbo supports Table sizes of up to 30 gigabytes, up to 16 million tables per database and 1000 active users. We estimate that the practical limit on database size would be <100 gigabytes.
INGRES	None. Hardware limitations only.

ORACLE Operating system specific.

Rdb/VMS None. A limit of 50 Gigabytes is practical because of current backup technologies. There is no real practical limit to the level of simultaneous usage as Rdb/VMS includes row level locking.

STARBASE No limits exist for either the size of the database or the level of simultaneous usage.

SYBASE Database size is limited only by available disc space. The number of simultaneous users is limited only by the operating system considerations (typical 512) and available memory. Each user requires 24 Kbytes of memory.

3.5.2 What file structure is used to support the relational database?

INGRES File structures listed (see table below) are in addition to secondary indexes.

ORACLE All data is stored in a set of operating systems files created by Oracle. The storage of data in these files is proprietary and different from traditional 3GL database structures.

Rdb/VMS Rdb/VMS maintains its own internal structure. It does not resemble any of those mentioned and is proprietary.
Rdb/VMS support's Codd's 12 rules including his "Zeroth Rule" which he starts off with but is not included in the 12, which states

> 'For any system that is advertised as, or claimed to be, a relational database management system, that system must be able to manage databases entirely through its relational capabilities'. (Appendix 1)

SYBASE The database imposes its own structures on raw disc devices and does not use the host system. Data is indexed internally using B-trees or B*-trees.

	Sequential	Indexed Sequential	VSAM or equivalent	Hashed Random (hashing on primary keys)	Other	
INFORMIX	-	-	x*	-	x	Informix Turbo uses RSAM
INGRES	x	x!	x#	x	x	Compressed & non-compressed versions of all.
ORACLE	-	-	-	-	x	See above
Rdb/VMS	-	-	-	-	x	See above
STARBASE	x	x	x	-	-	
SYBASE	-	-	-	-	x	See above

* C-ISAM in standard Database
! Full primary key used
ISAM and B*tree

VSAM The index structure of IBM's "Virtual Storage Access Method" is very similar to Knuths variation of the B-tree (the B+-tree). It includes various additional features such as the use of compression techniques.

RASM Informix proprietary file structure.

3.5.3 Which form of secondary indexing method is used?

Research into index performance suggests that certain developments of the B-tree (balanced tree) method give the best all-round performance. (refer to Part I, Chapter 1)

	B-tree specific to RDBMS	B*-tree specific to RDBMS	VSAM or equivalent	Hashed Random	Other	
INFORMIX	-	-	-	-	x	B+ tree
INGRES	x	x	x	x	x	!
ORACLE	-	x	-	-	-	
Rdb/VMS	x	-	-	x	-	#
STARBASE	x	-	-	-	x	Bit maps specific to RDBMS
SYBASE	x	x	-	-	-	

! All keyed storage structures may be used (or changed dynamically) for both secondary indexes & primary storage structure.

Secondary & primary indexes treated the same. Indexes may or may not allow duplicates, both B-tree and Hash indexes may be defined for the same field.

3.5.4 What datatypes are implemented

	Structured Text	Alphanumeric string (variable length)	Alphanumeric string (fixed length)	Floating point (single or double precision)	Digitised Image	Digitised Audio	Integer (1,2 & 4 byte)	Bit String	Date & Time	Other
INFORMIX	-	x+	x	x	x+	x+	x	x	x	-
INGRES	x!	x	x	x	~	~	x	~	x	-
ORACLE	x*	x	x	x	x#	x#	x	~	x	-
Rdb/VMS	x	x	x	x	x	x	x	-	x	-
STARBASE	x	x	x	x	x	x	x	-	x	Nulls, BLOB's
SYBASE	x'	x	x	x	x'	x'	x	x"	x	$

+ In release 4.0

! Word processing files (etc) may be pointed to by database tables

~ The ANSI string types permit the full range of byte values to be stored.
Bit string not directly supported, applications may support via full ASCII character set

* With ORACLE product SQL*TEXT RETRIEVAL.

Digitised data supported by use of raw data type, field limited to 65K bytes.

' Text & Image (or Audio) Datatype up to 2 Gigabytes per attribute.

" As for image or binary and varbinary.

$ Money (Double precision integer), Bit

3.5.5 Can compression be applied to the stored data?

INFORMIX	No	
INGRES	Yes	Tail-end compression of trailing blanks occurs. (Expect an end of year announcement of user-defined datatypes and functions.)
ORACLE	Yes	Oracle does not retain any redundant data - it does not store spaces or blanks in its records. Any other compression must be program driven through the user interface.
Rdb/VMS	Yes	Data compression is recommended for those tables not frequently accessed. It saves space and may reduce the time it takes Rdb/VMS to perform certain kinds of retrievals. It is optional and may be disabled. *(This is the only database to offer non-trivial data compression. Although the exact form is not stated, it is likely to be a form of Lempel-Ziv-Welch compression. See Part III: Image Compression)*
STARBASE	Yes	Automatic compression of nulls.
SYBASE	No	Compression/Decompression must take place externally to the database.

3.5.6 Can encryption be applied to the stored data?

INFORMIX	No	
INGRES	No	
ORACLE	Yes	Encryption of passwords and security information. (*Should be standard.*)
Rdb/VMS	No	
STARBASE	Yes	Has internal encryption and decryption facility.
SYBASE	No	Encrypt/Decrypt must take place externally to the database.

3.5.7 What levels of security are provided for the stored data?

INFORMIX	Permissions granted by the DBA control all access to the database.
INGRES	INGRES provides an external system to control user access and an internal system to protect against database corruption. The external security system is flexible allowing control authorised access to the database by user, date, time of day, location or a stored data value. Internally, INGRES is engineered to minimize single points of failure and provide maximum protection for the data.
ORACLE	ORACLE provides flexible security, allowing exact specification of the data each user is permitted to access or modify. ORACLE's security audit facility can trace all requests for data, giving detailed analyses of usage and the ability to track down unauthorized requests. Oracle is currently involved in a cooperative research and development effort with the US Department of Defence and the National Computer Security Centre

to define and develop further security enhancements. The result of this joint venture will be an Oracle product that most closely satisfies the requirements of the US defence and intelligence community.

Rdb/VMS The database file is protected by VMS file protection. The rights to perform database operations are kept in a set of Access Control Lists (ACLs), associated with entities in the database. ACLs are maintained by the owner of the database and may be updated at any time if you have the privilege. ACLs govern access to Relations, Views, Data definitions, Data manipulation operations, Database utility operations.

STARBASE Full vendor security, Starbase structured security, tailored, security through PowerHouse development. (Field, Row/Record/Table/File, Database).

SYBASE The Secure SQL Server has the ability to store data of multiple security classifications in a single database. Sybase has designed two versions of the server; the version designed at the B1 level runs under UNIX, and the version designed at the B2 level runs on bare hardware. Both versions provide mandatory and discretionary access control, auditing of security relevant events, and separate user administration roles.

3.5.8 What facilities are there to support databases that are distributed over more than one host computer?

If a database is to be distributed then it is important that this function is provided by a distributed DBMS rather than a network file system (NFS). It is much better to send the queries to the data than bring the data to the query. A DBMS with a heuristic optimiser will choose an intelligent accessing strategy and offer much better performance than a NFS-based distributed data manager.

INFORMIX Informix Net product allows for transparent distributed processing between local and remote databases. The next release (4.0) will allow multiple site read and single site update.

INGRES Distributed Processing (aka, remote access): PC LINK, INGRES/NET products. Distributed Database: INGRES/STAR product

ORACLE SQL*NET, Oracle's networking product which enables the location of data to be transparent to the user.

Rdb/VMS VAX Data distributor is used for distributing Rdb databases over more than one host computer (in this scenario, all computers must be VAX).

STARBASE StarBase StarNet, (Network server), StarBase StarGate, (SQL gateway).

SYBASE Sybase is an inherently distributed client/server product. Distributed update is supported via library routines which provide a two-phase commit service[1].

[1] Two phase commit is a second generation RDBMS feature. Only after all relevant updates have been confirmed is the data committed to the database. Starbase also uses two phase commit protocol.

3.6. RETRIEVAL FACILITIES

The interpretation of the questions in this section gives an indication of the products support. Starbase is part of a four component integrated product set which includes the PowerHouse 4GL. The PowerHouse 4GL can access PowerHouse StarBase, Rdb/VMS and RMS files directly through a protocol called Open Systems Relational Interface (OSRI), then through StarGate and StarNet can access any other SQL RDBMS.
The questionnaire from Rdb explicitly stated its interpretation ...

> " Rdb does not come pre-packaged with 4GL or graphical retrieval systems. It is purely a relational database management system. There are over a dozen third party interfaces as well as Digital supplied options. The questions asked will be answered in the context of Digital products (VAX Rally, VAX TEAMDATA and DECdecision) however this does not restrict the user to only these products."

From the answers of INGRES and ORACLE, the questions have been interpreted considering third party products available, while SYBASE (like Rdb) answered referring only to in-house software, without considering any third party products which exist. Thus for some questions answered in the negative by SYBASE the answers are the same as those given by the other products answered in the affirmative. In fact, one of the third party products that Sybase interfaces to is GemStone (Servio Logic Development Corp). Gemstone is the first commercially released multi-user Object Orientated Database Management System. The server software runs on a Sun3, Sun4 or Sun386i and the client may reside on a Sun Workstation, a Vax, an IBM PC, or a Macintosh. Gemstone uses an object orientated programming language OPAL and is well suited to applications which involve large volumes of highly complex data, including CAD, CAE, CASE, CIM, knowledge based systems or geographic information systems.

3.6.1 Is a 'Query-by-Example' facility provided?

Query-by-Example is a high level database management language that provides a convenient "fill-in-the-blanks" format in order to query, update, define and control a relational database. This allows a non-programmer to make relatively sophisticated queries without a knowledge of first-order-predicate calculus.

INFORMIX	Yes	
INGRES	Yes	QBF tool
ORACLE	Yes	SQL*QMX
Rdb/VMS	Yes	Rally
STARBASE	Yes	SQL/PowerHouse Quiz function/GDML
SYBASE	Yes	

3.6.2 Is a 'Query-by-Forms' facility provided?

INFORMIX	Yes	
INGRES	Yes	QBF tool
ORACLE	Yes	SQL*FORMS
Rdb/VMS	Yes	Rally
STARBASE	Yes	SQL/PowerHouse Quiz function/GDML
SYBASE	Yes	

3.6.3 Which user session aids are provided by the retrieval facilities?

	Switching to easier (faster) Query Mode or Language	Warning and Estimation of Long Query Completion	Saving of session files across an interruption	Saving of Selected resulted relations	Refinement of previous Queries	Macros of parts of Queries	Help Options
INFORMIX	-	-	-	x	-	-	x
INGRES	x!	~	x*	x	x#	x+	Full context sensitive Help, help on keys, field help
ORACLE	x	-	x	x	x	x	x
Rdb/VMS	?	?	?	?	?	x	x Answer incomplete
STARBASE	x	-	x	x	x	x	x Multiple
SYBASE	x!!	-	-	x	x	x	x

! Query Execution Plan may be saved automatically with 'repeated' clause.
~ Forthcoming Release
* Standard SQL 'transaction processing' features, automatic recovery process.
# Including QBF 'last Query' feature.
+ Views available, temporary tables etc.
? Rdb's incomplete answer implies that while all three packages provide macros and help other aids vary depending on the package.
!! VQL (Visual Query Language) is provided as a pick-and-point method to create SQL queries

3.6.4 Are there any graphical display options?

INFORMIX	No	
INGRES	Yes	
ORACLE	Yes	
Rdb/VMS	Yes	TEAMDATA and DECdecision
STARBASE	Yes	see PowerHouse Graphics
SYBASE	No	But the DB-Library interface is provided to allow access by third-party packages, eg: Dataviews (See SYNERGY document)

	Colour	Multiple Graphs on Same Axis	Stacked Histograms	Scatter Diagrams	Other Standard Features
INFORMIX	-	-	-	-	
INGRES	x	x	x	x	Pie Chart, Visual Programming - Intelligent Graphics editor.
ORACLE	x	x	x	x	Bit-mapped interface for some products eg X-windows
Rdb/VMS	x	-	x	x	Bar & Pie Charts
STARBASE	x	x	x	x!	Pie, Point, Stacked, Line, Horiz & Vert, Below Line, Multi Summary
SYBASE	-	-	-	-	See comment above

! Point Graphing

3.6.5 Are there any image display options?

INFORMIX No Not at present but maybe in the future.

INGRES Yes INGRES interface to 3rd Party product.

ORACLE Yes Oracle currently has a product in Beta, called Visuelle.

Rdb/VMS Yes Image data may be stored currently in Rdb. Digital's suite of image products due for release shortly will incorporate an interface to Rdb. This will include an image display option.

STARBASE Needs some clarification.

SYBASE No Images may be stored in the database using the "image" datatype. These images may be retrieved via the DB-Library interface for display by external routines.

3.6.6 Are there any examples of the DBMS being used for digital image storage (for example X-rays in hospitals)?

INFORMIX No

INGRES Yes One example is an art gallery owner in Sydney who has images of all his paintings stored in a database on his Sun.

ORACLE Yes There are many examples, GEOVISION, ARC-INFO, a VAR (Value Added Reseller) is FSTI (Forensic Science Technology International). (See Chapter 4, Section 2, Examples and Applications.)

Rdb/VMS Yes There are examples of Rdb/VMS being used for digitised voice and image. The time scale for this questionnaire did not allow details to be obtained.

STARBASE Yes BLOB (Basic Large Object) support of any digitised information is a standard feature.

SYBASE Yes The "image" datatype is a new feature and as yet there are no production applications but X-ray storage is typical of the applications for which this feature was designed.

It is interesting to note that most of the Oracle databases that are used to store imagery do not use the limited "raw data" database field type. Instead only a pointer to the image is stored in the database. Because of data field limit of 65Kbytes, Oracle suggested that using the datatype could actually be restrictive if there were future technology changes. In defence, additional benefits were cited by storing pointers to the image files rather than the files themselves. The most significant were for housekeeping and archiving. By using pointers, image files which are generally large in comparison with other database data can be kept separate, decreasing search times and making backups easier. However databases which support BLOBs also support storage features such as node and device allocation. Thus images may be allocated to a certain logical or physical device or even stored on a separate node. In addition BLOBs allow access and integrity of the image data to be handled by the database. There is a clear direction towards object orientated approaches for the integration of structured and unstructured data in the database [2.2]. This allows knowledge abstraction from the data. Direct derivation, manipulation or dynamic computation of image features become possible.

3.6.7 Can additional user functions be added to the language to provide derivation or computational facilities?

INFORMIX	No	
INGRES	Yes	Stored Database Procedures available now, user-definable Abstract Data Types available in a forthcoming release.
ORACLE	Yes	
Rdb/VMS	Yes	VAX Rally can call subroutines written in any native VAX language (includes those mentioned previously).
STARBASE	Yes	Triggers and/or PowerHouse.
SYBASE	No	But the SQL Server incorporates extensive mathematical functions so that quite complex routines can be encoded as stored procedures (SQL subroutines).

3.7. DATABASE ACCESS FROM THE APPLE MACINTOSH

Increasingly users will require transparent access to databases, remote print and directory servers from intelligent terminals and workstations such as the Macintosh. The range of capabilities available will increase with the emerging technology, however there are already a few solutions to the remote access of databases. All of the products listed have an application protocol interface (API) which resides on the Macintosh and sends SQL requests over the network to the host server, where the requests are evaluated and any results returned to the client.

3.7.1 Databases supported

PRODUCT	SUPPLIER	INFORMIX	INGRES	ORACLE	SYBASE	VAX/RDB	Other
CL/1	Apple	x	x	x	x	x	RMS files, DB2, SQL/DS
SequeLink	TechGnosis	-	x	x	x	x	
SQL*Net	Oracle	-	-	x	-	-	
SQLServices	DEC	-	-	-	-	x	

3.7.2 Transport Layers Supported

	Apple Talk	DECnet	TCP/IP
CL/1	x	x	x
SequeLink	x	x	x
SQL*Net	-	x	x
SQL Services	-	x	-

NB: Any database connection made using TCP/IP cannot guarantee username or password security, both of which must be validated by the host before any access is permitted.

CL/1

CL/1 (Connectivity Language 1) is an SQL based development tool for communication with databases. CL/1 is a proprietary language which is an extension of SQL. The SQL extensions include connectivity, non-relational database capabilities, error-mapping, and structured programming. Some examples of the Macintosh desktop applications that provide support for CL/1 are Hypercard, 4th Dimension, Omnis, Wingz, Dbase, and Excel. Once the connectivity language is learnt the application program written on the Macintosh can access a wide variety of host data bases without changing any code in the application. This has obvious advantages, but has the disadvantage that much of the work must be done on the server side.

CL/1 will be embedded in version 7 of the Macintosh operating system. While the VAX/VMS version has been in beta-test for some time, a release data for the product has yet to be announced.

SequeLink

SequeLink parses SQL statements using exactly the same syntax as would be entered if accessing the database directly from the host. There is no filtering of the SQL which allows the user to use any vendor-specific extensions provided by the host RDBMS. The client is resident on the Macintosh as a driver and the server is a network process on the host. The SequeLink driver on the Macintosh provides easy access from compiled languages such as Pascal or C, or through external functions to user-programmable interfaces such as Hypercard or 4th Dimension.

SQL*NET

SQL*Net comes with the networking version of Oracle for the Macintosh. This version is required for the Macintosh to access remote Oracle Databases. SQL*Net may be accessed by Hypercard or applications languages supported by MPW (Macintosh Programming Workshop). An interface to 4th Dimension (database) has also been announced.

The Oracle SQL request is either executed locally on the Oracle database or via SQL*Net on the remote database server. Like SequeLink the Oracle routines packetise the request and send it to the remote server. The server then executes the request and returns the result over the network to the client. The Oracle routines return the results to the calling application.

SQL Services

DEC have announced SQL services from VMS, Ultrix and MSDOS clients which will allow SQL requests to be executed on VAX and Ultrix servers. With the Apple and DEC cooperative agreement last year, a similar announcement is expected for the Macintosh. The structure is expected to be the same as that for SQL*Net and SequeLink, and provide access to VAX/Rdb/VMS databases.

4. Image Database Management Systems

4.1 Theory and Direction

The majority of work in database management has been on developing DBMS for traditional business applications, such as airline reservations and banking, that use well formatted or structured data. Even now, there is generally a belief that existing commercial DBMS are not well suited for efficient storage and manipulation of unstructured data, and that while the relational model is the most user orientated it does not satisfy the growing diverse requirements of the user community [2.3].

In the late seventies, early eighties the possibility of using relational databases for images began emerging[1]. During this time among the most notable examples were Chang and Fu who worked on the construction of relational databases from pictorial data (REDI) [2.4] and a relational query language 'Query-by-Pictorial-Example' [2.5]; which ran on a PDP 11/45 under UNIX with system software developed in C. A survey by Tamura [2.6] contains a review of the image databases in use up to 1983, including pointer access from conventional databases (ADM), approaches from image processing/ analysis systems (EIDES, MIDAS, IMDS, LIMS), GRAIN - originally a medical information system, and REDI. REDI (RElational Database system for Images) was designed for managing Landsat images and digitised maps. Most of these early systems were purely application orientated.

Recently, the combination of improved database design with decreasing hardware prices has realistically bought image databases into the fore. Thus the major objectives of image databases, previously hampered by slow response time, can be addressed: viz efficient storage, powerful retrieval capability and flexible data manipulation.

Image databases require the integration of structured and unstructured data, but still require imagery to be clearly distinguished for processing purposes. This integration can help in efficient processing of the unstructured data. Unstructured data can be managed efficiently by abstracting some knowledge from the data, storing it in a structured form, and then processing this subset of structured data. While it is widely advocated that the data structures should be object orientated, there is no indication of exactly how the images are best stored. Many data structures have been proposed. These include pixel orientated (see PICDMS [2.17]), quadtree based [2.7], R-tree [2.8], vector based [2.9], or 2-D string representations [2.10]. Only recently have more advanced query processing capabilities been emerging in scientific journals. These capabilities include precomputation and utilisation of spatial relationships, and dynamic computation of spatial relationships from the images. Integrated systems which have evolved to meet these needs are referred to as Image Understanding Environments [2.11] .

Object orientated approaches are promising for representing and manipulating unstructured data types. However, by themselves, object orientated approaches do not solve the problems of extracting and relating knowledge from unstructured data types and presenting

[1] In November 1981, an entire issue of IEEE Computer was devoted to Pictorial Information Systems.

data in a manner useful to human users of that data. Extracting information is time consuming, precomputation is space consuming. As far as image databases are concerned, while some software does exist it is largely at the developmental stage and has not filtered down from the scientific to the commercial arena.

4. Image Database Management Systems

4.2 Examples and Applications.

With the advent of cheap scanners, there are many examples of database systems incorporating images. Applications range from the use of dithered bitonals used to store non-ASCII data (large corporations include scanned photos in personnel files) to sophisticated image query and manipulation systems. Rather than being an extensive list this section is meant to give an overview of the work that is being carried out relating to image database management systems.

In Digital Review [2.12], a market research company Dataquest, estimates there are at least 25 full-text retrieval products on the market that "deal with a piece of paper as an image", and "image management systems are the next step".

One example is **MARS Series 3000 (Multi-user Archival and Retrieval System)** for the Apple Macintosh. MARS is currently restrictive because the niche the product aims for is broad high volume archival of records, such as contracts. It involves storage of binary information only - not manipulation of image data. Documents (text, hand writing, pictures, diagrams) are scanned at 200 dots per inch. Optical character recognition is done on the scanned documents which are stored as binary files, utilising CCIT IV compression (see Part 2, Appendix 1). Optical character recognition interprets pictures and diagrams as unrecognisable characters and thus any information contained in pictures or graphics can only be viewed on the original scanned document. With the introduction of WMRM discs onto the market, products such as MicroDynamic's MARS, currently based on 12" WORM optical discs with juke box capabilities, may be extended to offer image database systems.

PicturePower is an IBM compatible software/hardware package. It runs on IBM PS/2 or MS DOS compatible computers with VGA graphics and integrates with dBase III Plus. The package includes two cards, one for image digitising and display and the other for data compression. The system is designed to digitise and process images and enables the creation of image databases.

Astronomical Target Location Centre Data Base (ATLDB). This digital image database developed by A. Fresneau [2.13] at the Stellar Data Centre (Strasbourg) is to support the target identification and location functions of stars. Although the one year feasibility study should be complete (if started May 1987), no further papers appear to have been published.

The Library of Congress (USA) the world's largest repository of printed matter has successfully completed a pilot project in which an existing IBM based retrieval system (SCORPIA) has been interfaced with an optical disk image storage system [2.14]. It has successfully shown its ability to enable access to a variety of materials but the author is uncertain whether it will find a permanent place in public reading rooms or merely be a tool for library staff.

Forensic Science Technology Institute (FSTI inc S.A.) have created various image databases for specific applications. One example is for the South Australian Police Force red light camera system. At the Holden Hill Depot a (5 Mb) database was set up on a combination of PC's and μVaxes. Each set of photographs involved (two photos are taken to ensure an offence has been committed) is digitised and if necessary processed, using a standard set of image processing routines. An archival history is retained using Exabyte video 8 cartridges (2 Mb per tape). The database is Oracle, which was chosen for its mature architecture and large support base. Although Oracle can store image data in its fields, its record size is restricted to 65K and storing images incurs large I/O overheads in accessing. For these reasons only a pointer to the images is stored in the database record; however SQL* Forms handles the retrieval of images transparently to the user. Some minor image compression is used; the RGB images are packed from three bytes into two, decreasing image storage from 750 bytes to 512 bytes per image.

University of California Berkley (UCB) has designed a system called Imagequery, based on a Sun Workstation. It offers basic image processing capabilities integrated into a database of photographic images [2.15]. It is currently being ported to Apple's MAC II, IBM's PC RT, and DEC's MicroVax.

Empress [2.16] is an extensible object orientated RDBMS, designed for Sun networks. It provides a layered architecture with capabilities for load balancing and distributed databases over several nodes. An SQL-based language allows users to add data types, functions or operations. The 4GL, M-builder is bundled with the software. Empress is capable of handling unstructured data including graphics, images, voice and forms.

IMDAT [2.17] is a relational image database management system written in FORTRAN 77 and implemented on a VAX 11/750, with VMS operating system. Imdat uses a data dictionary to map logical values to the physical data. The logical values, stored as integers, represent raw data, images, feature vectors, character strings, etc. Images are obtained by referring to the data dictionary and then accessed through a linked list which is used for storage allocation. A query language Query-by-Procedure-Table (QPT) enables pattern recognition, similarity and structural retrieval.

PICDMS (Picture Database Management System) is a database management system for image processing and model-building. It is a grid orientated pictorial database management system designed and developed at ULCA [2.18], and commercialized through MIB Chock, CA, USA. A high level *tabular* language PICQUERY has been designed for PICDMS, as part of an architecture towards transparency between pictorial and non-pictorial databases. The range of image manipulation, pattern recognition, spatial and geometric operations, in addition to common, statistical, and user defined functions is very extensive. PICDMS is currently available for IBM PC and Prime computers, however it is said to be able to adapt to systems with a PL/1 compiler.

PSQL is an experimental query language for pictorial databases [2.19]. It is built on top of **ADMS** [2.20],[2.21], a high performance relational database management system. An extension of SQL, PSQL supports user-defined abstract datatypes. The database supports nonatomic non-zero space objects, direct and indirect spatial searches and direct spatial computation. In addition the database supports an advanced user interface. It allows direct graphics input specification and coordinates output display between the pictorial and alphanumeric data.

PART II: Database Management Systems

References and Bibliography

[2.1] Relational DBMSs, Xephon Buyers Guide, March 1986.

[2.2] M. Stonebraker, Inclusion of New Types in Relational Data Base Systems, Proceedings 2nd International Conference on Data Engineering, Los Angeles, CA, p 262-269, Feb 1986.

[2.3] A. Sheth, Managing and Intergrating Unstructured and Structured Data: Problems of Representation, Features and Abstraction, 1988 IEEE Fourth Conference On Data Engineering, Feb 1988.

[2.4] N.S. Chang; K.S. Fu, Picture query language for pictorial database systems, *Computer*, No 14, p 23-33, Nov 1981.

[2.5] N.S. Chang; K.S. Fu, Query-by-Pictorial-Example, *IEEE Transactions on Software Engineering*, Vol SE-6, No. 6, p 519-524, Nov 1980.

[2.6] H. Tamura; N. Yokoya, Image Database Systems: A Survey, *Pattern Recognition*, Vol 17, No 1, pp 29-43, 1984

[2.7] H. Samet, The Quadtree and Related Data Structures, *ACM Computer Surveys*, Vol 16, No 2, pp 187-260, Jun 1984.

[2.8] A. Guttman, R-trees: A Dynamic Index Structure for Spatial Searching, *Proceedings ACM-SIGMOD 1985 International Conference on the Management of data*, pp 484-492, Jun 1984.

[2.9] E. Jungert et al., Vega - A Geographical Information System, *Proceedings of the Scandinavian Research Conference on Geographical Information Systems*, Sweden, Jun 1985.

[2.10] S.K. Chang, C.W. Yan, D.C. Dimitroff, T. Arndt, An Intelligent Image Database System, *IEEE Transactions on Software Engineering*, Vol. 14, No. 5, pp 681-688, May 1988.

[2.11] D.T.Lawton, and C.C.McConnell, Image Understanding Environments, Proceedings of the IEEE, Vol 76, No 8, p 1034-1050, Aug 1988

[2.12] M. Levine, Going Beyond Text to Images is Next Step, *Digital Review*, Jan 9, 1989.

[2.13] A. Fresneau, ATLDB: A Global Data Base designed for Rapid Digital Image Retrieval. A Preliminary Design; *Bulletin Information Centre Donnes Stellaires*, May 1987.

[2.14] B. Manns; D. Wilder, Interfacing a mainframe database retrieval system with an optical disk storage system, 1987 IEEE Symposium in Mass Storage, p 7-11.

[2.15] H Besser, Image Processing Integrated into a Database of Photographic Images: Applications in Art, Architecture, & Geography, *Electronic Imaging '88, International Electronic Imaging Exposition and conference*, Vol 2, p 1214-1218, 1988

[2.16] L. Boyd, Empress Continues Reign over High-end Databases, *The Sun Observer*, p 42, Jun 1989.

[2.17] X. Zhang, J-K. Wu, W-T. Wan, Design of a Relational Image Database Management System: IMDAT, Proceedings of the International Workshop on Industrial Applications of Machine Vision and Machine Intelligence, p 310-314, 1987.

[2.18] T Joseph and A F Cardens, PICQUERY: A High Level Query Language for Pictorial Database Management, *IEEE Transactions on Software Engineering*, Vol 14, No 5, p 630-638, May 1988.

[2.19] N. Roussopoulos, C. Faloutsos, T. Sellis, An Efficient Pictorial Database System for PSQL, *IEEE Transactions on Software Engineering*, Vol. 14, No. 5, p 639-650, May 1988.

[2.20] N. Roussopoulos and H. Kang, Preliminary Design of ADMS : A Workstation-mainframe integrated Architecture for Database Management Systems, *Proceedings of the 12th International Conference VLDB*, Kyoto, Japan, Aug 1986.

[2.21] N. Roussopoulos, C. Faloutsos, T. Sellis, Overview of ADMS: A High Performance Database Management System, *Proceedings of the Fall Joint Computer Conference*, Oct 1987.

These final two references have appeared since this study's completion.

[2.22] F. Pascal, A Brave New World?, Byte, pp 147-256, Sep 1989.

[2.23] The Definitive VAX/RDBMS Guide to the Galaxy, ON$DECK Magazine, pp 37-56, Sep/Oct 1989. *The first DEC review since 1985.*

PART III: IMAGE COMPRESSION

Brevis esse laboro,
Obscurus fio.

Horace 65-8 B.C.
I strive to be brief and I become obscure.

THIS PAGE INTENTIONALLY LEFT BLANK

5. Image Compression

In 1985, NASA received 1 gigabyte of data from spacecraft daily. By the mid 1990's daily volumes from space-based payloads will reach 2,400 gigabytes (2.4 TB). This statistic is even more sobering considering that in the 28 years to date less than 5 TB of data have been collected. The accelerating requirements of NASA will be a driving force for high commercial interest in better data compression techniques [3.1]. Greater emphasis will be given to overcoming the problems that must be solved before the widespread use of automatic data compression becomes feasible.

Compression strategies will vary with the types of data to be stored as well as the intended uses. The types of redundancy vary with the types of data. Unlike most imagery text has little local character repetition or spatial correlation but is more likely to benefit from the skewness of character distribution or high-usage patterns. Different tradeoffs must be considered in order to choose the best compression techniques for text, floating point numbers, or imagery. This report focuses on the requirement of data storage and compression of digital imagery. This still presents a complex spectrum of choices.

Compression techniques may be lossless and invertible, or lossy and non-invertible. The specific compression requirements for any given imagery determine the compression schemes available for that data.

Off-line backups for long-term data storage may be best stored in a predictable amount of space regardless of image content. Such schemes use the spatial dimensions and bit depth of an image to determine the space required. This enables the image to be accessed randomly, allowing users to window portions of large multispectral satellite imagery.

If the primary consideration is the *most efficient* storage method without loss of data integrity, then a lossless compression technique is used. The data remains invertible and allows better usage of available space by taking advantage of spectral, spatial and/or temporal correlations. For example, multispectral images may have correlations between the bands or correlations between neighbouring pixels. Lossless data compression can reduce space requirements by a factor of about three.

More stringent space requirements may force a loss of original data integrity. In this case the images stored retain only enough information for a particular application. An example may be a ship database where an image header contains the ship class (which is retrieved as the template) and the associated data is a minimum set of differences required to uniquely define the ship within the class. If imagery is to be stored for viewing only, then much higher compression ratios can be tolerated than if imagery is to be processed further, or used for detailed identification or analysis. A large online database may contain highly compressed representations of imagery for screening purposes. At the farthest extreme what is retained is not an image at all, merely a set of descriptions.

Image compression rates are generally expressed as a compression ratio or absolute storage requirement (ie bits/pixel). Other considerations are the compression and decompression time and for image transmission, the image transfer ratio. The compression ratio is the ratio of the original image size to the compressed image size. The image transfer ratio is the product of the compression ratio (plus any overhead information) and the data transfer rate. If the image file is compressed, the total transfer time will be reduced, however the decompression time must be added. If the decompression is done in software it may take more time than was saved by reducing the transfer time. This is particularly true if recoverability is important. Non-invertible compression algorithms give higher compression ratios, but result in loss of data integrity. However, for some applications, compression which degrades the signal no more than the existing noise may be considered lossless with respect to the information content. In fact, some applications may benefit from compression, if it is information extractive or performs the initial steps of processing. For example hierarchical coding may be utilised for image matching; transform coding may be utilised to reduce noise artifacts.

The basic requirement of data compression is that it produce a representation of the original data with a reduced data set. Measuring the effectiveness becomes more difficult when the compression is not lossless. Evaluation techniques are often based on simplicity rather than scientific merit, or actual information content. The value of mean-square or root-mean-square calculations or added noise calculations depend on the application. For example, complex images retain a higher visual quality for a given Normalised Mean Square Error. Normalised Mean Square Error or MSE is also highly dependent on artifacts produced by compression techniques, increasing the difficulties of meaningful comparisons. An alternative measure of data quality is the Haussdorf Image Distance [3.2]. This integrates both spatial and spectral (between bands) distortion into one single measure of error. It is widely used in methods based on compression using fractals; its value in other techniques varies. Ultimately, an improved error metric is required.

Most first generation image compression techniques involve some form of predictive coding or transform coding [3.3], [3.4]. (Some methods only fit either category very loosely and there are certainly other methods that fall outside of them completely.) Predictive coding uses value(s) of earlier pixels/lines/frames to form a prediction for the present data. Transform coding uses values of all pixels. It is generally less sensitive to errors and produces higher performance but involves a more complex implementation than does predictive coding.

High correlation within an image allows a high degree of compression. Non-adaptive systems, in particular, may be expected to produce poor results where image properties depart significantly from the norm, whereas any image compression technique will work well when there is high degree of correlation within an image.

Predictive coding and most forms of transform coding, are purely statistical in nature - that is, they make no attempt to abstract features the human observer may consider useful, or code in a way that is meaningful. By dealing with statistical redundancy the coding methods generally remain reversible. Within information theory and coding theory these methods reach a limit at compression ratios of around 10:1. Coding systems matching the human visual system have been the focus for some recent second generation compression techniques [3.5]. Second generation techniques use subjective redundancy for feature abstraction. Subjective redundancy implies by its nature, irreversibility, but does allow higher compressions.

5.1 Image Compression Techniques

Many coding systems are combinations of different schemes, thus this section is loosely grouped under various general categories. The coding systems under each general heading include fixed, adaptive and combinatory methods. Adaptive data compression techniques are useful when the statistics of the data are not known in advance, or are changing slowly. If the image is not homogeneous and its redundancy characteristics vary within the image, then compression efficiency declines if the variation significantly exceeds the adaptive range of the compression implementation - the effect of "block size" on compression is not trivial.

Full-Raster

An uncompressed or full-raster image is stored in a fixed amount of storage independent of the content of the image. Typically the grey level[1] of each pixel in the image is stored systematically row by row, left to right starting at the top left corner of the image and ending at the bottom right corner. A standard image[2] requires 8-bits/pixel when stored in this way.

Universal Coding Algorithms

Universal Coding Algorithms assume the statistics of the data are not initially known by the coder. These methods attempt to measure the statistics during the actual coding operation and adapt themselves to maximise compression. At the simplest level they are adept at exploiting repetitious pixel patterns, but are not so good at taking advantage of correlations between adjacent lines, fields, or frames.

Run-length Encoding scans the image in 1 dimension and replaces strings of consecutive pixels with identical grey level values by the number of pixels in the run and the grey level of the pixels. The string may have a variable or maximum length. If constant length run-length encoding is used this scheme requires 1 byte for the number of consecutive pixels at that level and 1 byte for the grey level. To store a worst-case standard image with every pixel different would require 16-bits/pixel, twice the amount required by the full-raster method.

An adaptive compression technique was developed by Lempel and Ziv while investigating a complexity measure for strings [3.6]. The **Adaptive Lempel-Ziv** coding technique based on string matching, has been extended by Welch into the Lempel-Ziv-Welch algorithm [3.7]. This is the coding method used in the standard UNIX "compress" command.

The LZW algorithm is based on a string translation table that maps input characters into fixed length codes. The LZW string table contains strings previously encountered in the message being compressed. In [3.8] Welch uses the LZW procedure on text, object code, source codes, floating point arrays and formatted scientific data (but not imagery) and achieved consistent compression ratios of around 2 for each of these. He advocates the use of this algorithm because it is adaptive yet simple, permitting high speed execution.

The original **Huffman Coding Technique** [3.9] was based on the varying frequency of characters within a character set. The standard Huffman encoding procedure assigns codes to input symbols such that each symbol in the code was approximately $\log_2(\textit{symbol probability})$ bits, where *symbol probability* is the relative frequency of the occurrence of a given symbol (expressed as a probability). The most frequent characters are assigned shorter codes and the longer codes are constructed so that the shorter codes do not appear as prefixes. For a skewed frequency distribution this minimizes the mean number of bits per character. A variation is the adaptive Huffman's encoding method which changes the codes as the frequency of the characters change.

Huffman coding is often used in conjunction with other coding methods [3.10]. A modified Huffman code has been chosen as a CCITT standard (see Appendix 2).

---

[1] The term 'grey level' typically refers to one of the 256 possible indices stored in 8-bits which can be mapped through an index table (lookup table) to produce a grey scale image.

[2] For this study, a standard image refers to an image size of 512 by 512 = 256K pixels with 8-bits (1 byte) per pixel.

Predictive Data Compression Techniques

A major objective of predictive coding is to produce a differential signal that is small on average. The better the predictor the smaller the entropy and the lower the bit rate required for coding. Block coding can easily be applied to the differential signals. For **Delta Encoding** (or **Gap Compression**), the difference in grey level (the delta value) between two consecutive pixels is encoded rather than the pixel values themselves. Either the delta value is stored in the encoded image or a code value represents the delta. Non-linear delta encoding is generally better than linear delta encoding, however both suffer from edge distortion [3.11]. In **Differential Encoding**, the difference between an image and a template or previous image, is stored.

Sometimes predictive coding is used to specifically refer to **Differential Pulse Code Modulation** (DPCM) [3.12]-[3.14]. Pulse Code Modulation refers to analog to digital conversion. It has been used as a video digitising scheme for the purposes of transmission and storage since 1951. In DPCM an attempt is made to predict the pixel to be coded. DPCM is a method of scalar quantization which exploits the interpixel dependencies and then quantizes the resultant prediction error. Adaptive Differential Pulse Code Modulation can provide a lossless compression of approximately 3:1 for imagery. It is attractive because it provides a fast software implementation for lossless compression. By incurring some data loss the compression ratio can increased to around 5:1 [3.15].

Cheung [3.16] provides a recent example of predictive coding. A number of neighbouring pixel differences are weighted linearly to predict a pixel. The method is intended for use with video conferencing, where the interframe variation is generally quite small. Cheung achieved data reductions from the original 8-bits/pixel to under 0.5 bits/pixel when spatial correlation was used. When frame differencing was also used the coding requirements were reduced to about 0.25 bits/pixel. These compression rates were achieved with an image degradation of 0.2% Mean Square Error, which does not mean a lot out of context but probably retains moderate to good images for viewing. Higher compression ratios are accepted more easily if an image is only to be viewed.

Hierarchical Encoding methods

Run-length encodings can give good compression ratios, but contain no further structure, making them difficult to manipulate. Hierarchical data structures are based on the principle of recursive decomposition of space. Thus they retain spatial proximity in the file organization. This allows sampling of an image in 'variable resolution' which may be useful for spatial efficiency, high speed matching, edge detection, or segmentation. Hierarchical encoding methods range from general data driven structures of sequential hierarchical decomposition with no restrictions on segment shape [3.18] to region quadtrees.

A **Quadtree** is created by dividing an image into quadrants and then further sub-dividing each quadrant into subquadrants until each subquadrant is of uniform grey level, or has a variance within a given variance threshold [3.19]. For binary images sub-division continues until each block of data consists entirely of 0's or 1's, which may be at the single pixel level. The root of the tree represents the entire image. Each node of the tree represents either a leaf node or has four sub-nodes. Thus the quadtree can be characterised as a variable resolution data structure. The disadvantage of using quadtrees is that they are dependent on the positioning of the origin. However, this is mainly a problem with simple images. Most complex images would not result in large improvements in storage due to optimum placement of the origin.

The original formulation of the quadtree encodes it using pointers. To reduce the space pointerless approaches exist [3.20]. With an efficient pointerless coding scheme, quadtrees require about 14 percent overhead and store a worst-case standard image with every pixel different in 300 Kbytes (9.375 bits/pixel) [3.21].

Quadtree structures have been used for storage in geographic information systems [3.22]. However, Rosenburg maintains that quadtrees are overrated for such applications [3.23]. He compares linked lists, quadtrees, and multidimensional binary trees (***k-d* trees**) as structures that support fast region searches in 2-space. Although he does not use pointerless quadtrees, he noted that quadtrees of higher and higher threshold values use less and less memory, approaching linked lists in the limit. *K-d* trees are more expensive in terms of memory than quadtrees. He found they also tended to get deeper, be slightly less balanced and take longer to build than quadtrees. However he felt that the performance of *k-d* trees in region searching, particularly for point searches was so much better than quadtrees that the advantages outweighed any disadvantages. He advocated using *k-d* trees for any area operations on large, geographically-organized databases.

Variations of the quadtree method include linear quadtrees [3.24], P-compressed quadtrees [3.25], pyramid structures, 2D-trees [3.26], octrees (3D quadtrees) and more [3.27]. How appropriate these methods are depends on whether additional processing or manipulation is required. Hierarchical data structures are attractive because of their conceptual clarity and ease of implementation. Hierachical structuring methods exploit features of human perception which find application in second generation coding methods.

Transform Coding

Instead of coding the image as discrete intensity values over a set of sampling points, an alternative representation is made by linearly transforming blocks of pixels into blocks of data called coefficients and then quantizing the coefficients selected for storage. Each block of N pixels is transformed (usually by a linear orthonormal matrix) into a block of transform coefficients, and the insignificant coefficients are discarded. The remaining $M=pN$ ($p<1$) coefficients are then coded and stored until required, when the inverse operation takes place. This can be interpreted as discarding "high frequency" noise.

Adaption either changes the transformation to match image statistics or changes the criterion for selection and quantization of the coefficients to match the subjective quality requirements. This can increase coding efficiency by about 25-30 perecent. Many such transformations have been used, for example the simple Hadamard transform, the Discrete Cosine transform or the more general Karhunen-Loeve.

The **Walsh-Hadamard Transform** uses a symmetric linear orthonormal transform matrix which can be described recursively. An advantage of the WHT is that most of the computation only requires addition and subtraction, whereas most other transforms require multiplication as well. The fast 2D Hadamard Transform will typically provide a compression ratio of 8:1 for images. Conceptually, the 2D Hadamard transform and its results are similar to quadtrees.

In recent years, the **Discrete Cosine Transform** has become the most widely used of the transforms that conserve image energy or mean-square-values (unitary transforms[1]). Under certain circumstances its performance is close to the Karhunen-Loeve. Cosine transformations are popular because they appear to be well matched to the statistics of general picture signals. Depending on the cost/performance considerations, one, two, and three dimensional (two

---

[1] Any linear orthonormal transform is unitray. A Unitary Transform is one s.t. $T^{-1} = T'$, where ' denotes transpose.

spatial dimensions and time) blocks have been used for transformation. Using a fast 2D Discrete Cosine transform compression ratios of 12:1 are achievable. With a Digital Signal Processor with 5 MIPS integer arithmetric this computationally intensive method (16x16 FDCT) is reduced to 15 seconds for a standard image [3.31].

Dinstein [3.32] uses a DCT coder with adaptively optimised block sizes. Each rectangular block may have width or length of size 8, 16, or 32. The blocks are clustered from an original size of 8x8. Optimisation is achieved by applying a technique from graph theory. This method increases fine detail preservation and decreases the blocking effect generally evident. It achieves a compressed bit rate of about 0.5 bits/pixel, but also involves some overhead information that must be transmitted in order to reconstruct the compressed image (suggesting a compression ratio of less than 16:1).

The **Karhunen-Loeve Transform** (or **Principle Components Analysis**) is also an orthonormal linear transformation. The KLT has coefficients that are uncorrelated, and their mean square values (MSV) are equal to their respective eigenvalue. If the KLT basis vectors are ordered according to decreasing eigenvalues, then a compaction of energy into the lower index transform coefficients will result. Using the mean square error distortion criterion, the KLT achieves the best energy compaction of any linear transform [3.28]. However discarding the insignificant coefficients is not always trivial, or even appropriate. Landsat MSS has 4 bands (bands 4,5,6,7). When transformed using principal components (Karhunen-Loeve Transform) the 4^{th} band contains of the order of 0.1% of the original image information. This information is completely uncorrelated with the other bands and if it is just noise (as is often the case) can easily be discarded reducing the the original storage requirement by 25%. For some agricultural purposes retaining the first two components is enough. However this 4^{th} component may also contain other information such as smoke, or small road detail, which may be valued, implying the 4^{th} components should be kept. Comparable results occur for Landsat Thematic Mapper, or Spot data, however the effective compression ratio is unlikely to be more than 2 [3.29],[3.30].

As well as just being used to spectrally compress multiple bands at each pixel, KLT could also be used to spatially compress blocks of pixels. Whereas other unitary transforms on a block of pixels, such as Fourier or Walsh-Hadamard use a predefined set of basis functions (such as 0's and 1's or sines and cosines), the KLT would effectively find the optimal set of basis functions for representing each block. Very large images would be needed to find this basis set for compressing even moderate sized image blocks (eg 4x4), since the size of the data cloud needed for accurate principle components analysis grows exponentially with the dimensions of the cloud. Nevertheless this technique could conceivably be used on satellite data, given the amount that is currently available.

In **Hybrid Transform Coding** linear transformations of blocks of pixels is followed by predictive coding of the resulting coefficients based on previously transmitted adjacent (spatially and temporally) blocks. Combining DPCM and transform coding techniques can achieve compressions of around 8:1.

Transform coding is criticised for its poor ability in coding edges. Edge loss is the archetypical result of loss of high frequency components.

Vector Quantization Methods

Vector Quantization [3.33]-[3.35] describes the class of methods which directly code blocks of pixels rather than encoding individual pixel values. There are many forms of vector quantization including lattice, finite state, classified, product, predictive, and hierarchical. An image is divided into subimages, usually of size 4x4. These subimages are compared with a set of predetermined subimages (code vectors) in a 'codebook'. The codebook entry which is closest to the subimage is selected according to some measure and the correspoinding codebook address is transmitted.

In practically all the recent studies of vector quantization the mean-square-error (MSE) has been used exclusively to choose the partitioning and code vectors to minimize the overall distortion for the data. The design of image vector quantizers generally requires a set of training images to create the codebooks to encode or decode the data. The vector quantization code vectors and partitioning are determined iteratively by repeated processing of the training set. Adaptive vector quantization simultaneously designs the codebooks as the data is being encoded or quantized. The codebook provides a simple "receiver structure" which is the main advantage of vector quantization. However codebook searching can be prohibitively complex and lengthy, as can the process of creating codebooks.

Vector quantization does not provide lossless compression. In addition to the amount of interpixel correlation, the quality achievable depends on the block size of pixel vectors and the suitability of the codebook to the images being coded. The loss on the training images is always considerably smaller than the loss on pictures outside the training set. Images with compression rates of greater than 10 provide acceptable compression for video conferencing and provide images of adequate quality for medical diagnosis [3.36].

Neural Nets

Although there has been a great deal of recent interest in neural networks, a literature search found very little evidence of their application for data compression. Neural networks are designed to work where there is little *a priori* knowledge but a large amount of training data. A neural network can be used to obtain an internal representation of image features or on any method that will benefit from parallelism such as vector quantisation. A counterpropagation network can be used in hierarchical manner to achieve more compression than would be achieved with a single network. A counterpropagation network is described to function as a "statistically optimal self-adapting look-up table" [3.37]. Hecht-Nielsen proposes the use of a counterpropagation network to carry out vector quantisation data compression, but does not give any examples of the compression possible using this technique.

A neural network designed to compress coherent images using a diffusion equation is given in [3.38]. It follows a maximum entropy approach that seeks a set of features which is maximally correlated with the next pixel value, but at the same time has minimum internal redundancy. Examples with compression factors of 2, 4 and 8 are shown on the speckled images of SAR data but these are not lossless and only compression factors 2 and 4 fairly accurately represent the original data. No measures of representation are given.

Fractals

Naturally ocurring structures look complex, but are far from random. The underlying natural laws have simplicity of form. Fractals extend the simple Euclidean geometric descriptors from points and lines, to enable constructions that resemble the products of natural laws. Fractals [3.2], [3.39], [3.40] are geometric or data structures which retain the complexity of their structure under magnification. Compression ratios of 10,000:1 have been achieved, however the resulting image looks more like an artistic representation of the original than an exact copy [3.41]. The larger an image and the higher the resolution, the more it can be compressed. Obviously expecting such high compression ratios on an image 256 x 256 would be unrealistic, since it would mean storing the entire image in $6^1/_2$ pixels.

A fractal can describe much or all of the information in an image (if it has a natural homogeneity) in terms of a few succinct rules. These rules are a series of affine transformations (linear plus translation) each of which is assigned an associated probability. Fractal image compression is iterative and as such is computationally intensive in nature, requiring multiplications and accumulations to build up an image. However the rules used consist of low dimensional matrix transformations which are conducive to high speed hardware implementations, especaily those involving parallel processing.

Fractals do contain structure and allow some image analysis to be done on the compressed data. Matching or similar textures may be found, objects identified and classified. However, further research must be done to formulate consistent methods of relating the parameters used in the compression to aspects of the original image.

A system currently available for the Sun workstation called VRIFS (see section 5.2) is specifically designed for biological modelling. With a compression ratio of 100 to 1, an Xray that is a high quality approximation of the original would involve approximately 100 contractive mappings. Using an AMT Distributed Array Processor, the fractal transform can compress (or restore) a 256x256 binary image in under 10 seconds [3.42].

In a sequence the rules and images change only slightly from one image to the next. Fractals allow low computational complexity of algorithms, thus by making small changes in code an animated sequence can be created. This makes fractals attractive for applications requiring real-time implementation. Standard methods incorporated with fractals are being used to search for solutions in the new lucrative field of high definition TV transmission where restrictive bandwidth regulations (in the US) will not allow lossless compression [3.42]. Goel [3.43] uses fractals to modify the run-length encoding method from the original lossless form. Fractal geometry aids the coding process by allowing certain variation in pixel values to increase the run length to a maximum. The amount of variation allowed depends on the complexity of activity in the scene. Humans find it easier to notice an object in a uniform background than one in an active scene. Edges are the most important perceptual component of any image and so are encoded as actual values with run-length zero. This method claims good quality reconstructed colour images at 1.25 bits/pixel.

Apart from the speed constraint, fractals seem to have specific potential within database and GIS applications for generating natural scenes of clouds and trees, for example for flight simulations.

Second Generation Image Coding Techniques

First generation techniques are limited by the simple descriptions of structure they use, usually in terms of first and second order statistics and interpixel redundances. Second generation techniques use features of human perception to exploit structure, for example decomposing the image into several components and quantizing each component separately.

In [3.5] Kunt et al. give an excellent introduction to second generation image coding techniques. They show two examples of region merging to obtain contours and texture coding using a two-dimensional polynomial function with added microtexture. Both ex^mples have attained compression ratios of 50:1. In order to obtain better edges Kunt et al. use directional decomposition based coding. An eight component directional filter is used to record the edges, while the low-pass image is stored using transform coding. The improvements in image quality obtained by this method are computationally expensive. However Kunt felt that these methods do not produce as high quality images as two other local operator based techniques presented in the same paper. One was a pyramid coding technique, and the other was called anisotropic nonstationary predictive coding; both are hybrid methods combining predictive and transform coding.

Contour Coding or **Region and Texture Coding** is the most common example of a second generation technique [3.4],[3.5],[3.43]-[3.45]. An image is separated into two parts. A binary image containing the high contrast boundaries and the remainder. The contours are run-length encoded and the rest of the image which contains only low frequency and texture information can be coded using predictive or transform techniques, or fractals. Contours may be extracted using edge detection or region growing techniques. Contours obtained using edge detection are not necessarily closed and may lead to some problems in coding textures. Both methods can contain borders that are not necessarily edges in the images.

The use of **Multiple Scales** of image representation is known to occur in the human visual system. Todd [3.46] uses a recursive linear multi-resolution model which also makes use of the orientation selective properties of the visual cortex. This method is shown to be effective at rates below 1bit/pixel, and capable of producing decoded images at compression ratios of up to 100, without the 'blocky' structure common in pyramid techniques. The new theory of **Wavelets** is a generalisation of pyramids. It combines the underlying ideas of resolution on many scales from pyramids with Fourier transform techniques [3.50].

The method of **Convex Projection Onto Sets** [3.47] is based on regarding a quantized compressed image as defining a set of images which includes the original image, rather than uniquely defining a distorted image. Vector quantization is combined with a form of Transform coding which quantizes the phase component. The set of convex sets chosen for this method of convex projection is computationally expensive and has achieved compression rates of less than 1 bit/pixel. The two images shown in [3.45] demonstrate compression ratios of 9.8 and 9.0.

5.2 Software for Image Compression

The growing importance of image compression for efficient data storage can be seen in the answers given to question 5.5 of the market survey. Until recently [2.1], no relational database management system used any form of data compression. The existence of the standard UNIX "compress" command which uses adaptive Ziv-Lempel-Welch coding is an indication of the acceptance and that the benefits of compression are beginning to be recognised. In addition to the systems in Part II section 4 which used compression in image databases, the following are two packages available specifically for image compression.

Tools for Image Compression (TIC) - is a menu driven software package which has been implemented on a Sun workstation, under the UNIX operating system. It assumes *a priori* knowledge about properties of compression and the images to be compressed. A variety of compression techniques are provided; Huffman code, run-length, differential, quadtrees, adaptive hierarchical and binquad encoding. The ability to add new methods is also provided [3.46].

Vector Recurrent Iterated Function System (VRIFS) - is a highly interactive menu driven software package for a Sun workstation. The core algorithm generates natural scenes, manmade objects, fractal entities, standard graphics, fonts, maps and diverse textures. A feature known as multi-scale fusion allows the user to depict image reality in various resolutions and magnifications.
VRIFS Presents - is a colour display package which interfaces with VRIFS for multi-resolution presentation of the synthesized graphic image.

Fractal-Image Analysis System (F-IAS) - is an analytical tool which works with two-dimensional digitised images at 512 x 512 x 6 bits/pixel. Image enhancement features include background subtraction, zoom, histogram modification (equalisation, contrast enhancement, thresholding), filtering for smoothing/sharpening images, touch-up painting, boundary detection, and tracking. Measurements include fractal dimensions, areas, perimeters, lengths, numbers of objects, object locations, intensity histograms and averages.
The cost of VRIFS is approximately US$ 40,000, from Iterated Systems, Inc.

5.3 Hardware for Image Compression

In time, the problem of choosing and implementing an appropriate compression routine will be taken out the of user's hands as custom VLSI chips become a standard component of communications and data storage. The compression method that is most often cited for use in these chips is some form of vector quantisation [3.47].

A variety of other hardware solutions exist depending on the computer platform. For example, on the Sun Workstation there is a monochrome image compression/decompression board (MP-152). The processing module provides CCITT Group III 1-D/2-D and Group IV compression and decompression (see Appendix 2).

5.4 Conclusion

Decreasing storage costs and increasing read/write speeds and available bandwidth will not be enough to offset the huge increase in available data over the next ten years. The importance of efficient storage is apparent by the growing commercial interest in hardware and software compression solutions.

However, there remain considerations that must be addressed. An important issue is that blocks of compressed data generally do not have predictable sizes, presenting storage management problems. For most compression techniques, the size of a compressed image is unpredictable and depends on its content. For lossless compression there is no assurance the data will compress at all. In fact, the image may require more space in its "compressed" form. Thus if space is to be preallocated, it must be at least as big as the original image. Processing the image may alter only a few pixels but significantly increase the compressed image size. Therefore it may be unwise to compress frequently-used or temporary images.

The reluctance of users to lose any information means that lossless data compression methods are more likely to be generally accepted. However if the choice of lossless compression were optional and required an understanding of compression methods, it would not gain general acceptance. Therefore, any method should be simple to implement, or be implemented automatically. While specific types of imagery, such as SAR, have unique characteristics and may benefit from specific data compression methods, it is probably more effective to use conventional data compression methods for all image data.

Ideally the most desirable compression technique is one that dynamically adapts to the redundancy characteristics of the data and achieves high compression rates with low overheads. Compression ratios of around 10 using conventional techniques, are acceptable for images intended for viewing and as such may be adequate for some database applications, such as preliminary searching, but are not recommended for images intended for applications such as image processing and analysis. A method providing a fast lossless software solution such as adaptive DPCM, which gives consistent compression ratios of around 3, is probably the most desirable compression scheme for image processing and analysis applications. Scene generation and facet modelling applications could benefit from the second generation techniques. For these applications the initial generation times are not crucial and the high compression ratios retain adequate information with low space requirements.

PART III: IMAGE COMPRESSION

References and Bibliography

[3.1] L.S. Kempster and J.B. Martin, In Search of: NASA space data storage solutions, *1987 IEEE Symposium on Mass Storage Systems*, pp 27-31, 1987.

[3.2] M.F. Barnsley and A.D. Sloan, Fractal Image Compression, *NASA - Proceedings of the Scientific Data Compression Workshop*, pp 351-365, May 1988.

[3.3] A.K. Jain, Image Data Compression, *Proceedings of the IEEE*, Vol 69, No 3, pp 349-389, Mar 1981.

[3.4] A.N. Netravali and B.G. Haskell, *Digital Picture Representation and Compression*, Plenum Press, New York, pp 147-244, 1988.

[3.5] M. Kunt, A. Ikonomopoulos, and M. Kocher, Second-Generation Image-Coding Techniques, *Proceedings of the IEEE*, Vol 73, No 4, April 1985.

[3.6] A. Blumer, Noiseless Compression using Non-Markov Models, *NASA - Proceedings of the Scientific Data Compression Workshop*, NASA conference Publication 3025, pp 367-376, May 1988.

[3.7] A.N. Netravali and B.G. Haskell, *Digital Picture Representation and Compression*, Plenum Press, New York, pp 163-164, 1988.

[3.8] T.A.Welch, A Technique for High-Performance Data Compression, *Computer*, pp 8-19, June 1984.

[3.9] D. Huffman, A method for the Construction of Minimum Redundancy Codes, *Proceedings of the IRE*, Vol 40, pp 1098-1101, 1952.

[3.10] G. Benelli, V. Cappellini, E. Del Re, R.Fantacci, Image Data Compression using the Laplacian Pyramid with Adaptive Huffman Encoding, *Advances in Image Processing and Pattern Recognition, Proceedings of the International Conference*, pp 229-233, 1986.

[3.11] S. Ropiequet (Ed.), Optical Publishing - A Practical Approach to Developing CD ROM Applications, *CD ROM Volume 2*, Microsoft Press, Washington, pp 156-158, 1986.

[3.12] T. Taniguchi, K. Kamikura, S. Itoh, T. Utsunomiya, Variable-Length-Code-Selective Reversible Predictive Coding for Multilevel Images, *Electronics and Communications in Japan*, Part 1, Vol. 71, No. 10, pp 102-113, 1988.

[3.13] A. Hayashi, DPCM_PCM Hybrid Coding and Its Properties, *Electronics and Communications in Japan*, Part 1, Vol. 71, No. 10, pp 56-63, 1988.

[3.14] S.B. Ng, L Schiff, Two-Tier DPCM Codec for Videoconferencing, *IEEE transactions on Communications*, Vol 73, No. 4, pp 380-386, April 1989.

[3.15] D.R. Ahlgren, J. Crosby, D. Erigat, Compression of digitised images for transmission and storage applications, *SPIE Image Processing, Analysis, Measurement, and Quality*, Vol 901, pp 105-113, 1988.

[3.16] W. N. Cheung, Spatial and Temporal Prediction of Picture Elements using Pixel Differences, *Proceedings of Australian Symposium on Signal Processing and Applications*, Vol 1, pp 65-68, April 1989.

[3.17] M. Lech, D. Nandagopal and D.A.H. Johnson, Two Dimensional Linear Predictive Coding for Images, *Proceedings of Australian Symposium on Signal Processing and Applications*, Vol 1, pp 73-77, April 1989.

[3.18] J.-M. Beaulieu and M. Goldberg, Hierarchy in Picture Segmentation: A Stepwise Optimization Approach, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 11. No. 2, pp 150-162, Feb 1989.

[3.19] H. Samat, Data Structures for Quadtree Approximation and Compression, *Communications of the ACM*, Vol 28, No 9, pp 973-993, Sept 1985.

[3.20] E. Kawaguchi and T. Endo, On a method of binary picture picture representation and its application to data compression, *IEEE Transactions on Pattern Analysis and Machine Intelligence 2*, No. 1, pp 27-35, Jan 1980

[3.21] D.L. Lansing, Experiments in Encoding Multilevel Images as Quadtrees, *NASA Technical Paper 2722*, Sept 1987.

[3.22] H. Samat, A. Rosenfeld, C.A. Shaffer, R.E. Webber, A Geographic Information System using Quadtrees, *Pattern Recognition*, Vol 17, No 6, pp 647-65, Nov/Dec 1984.

[3.23] J.B. Rosenburg, Geographical Data Structures Compared: A study of Data Structures Supporting Region Queries, *IEEE Transactions on Computer-Aided Design*, Vol CAD-4, No 1,p 53-67, Jan 1985.
(Rosenburg includes the routines (in C) for quadtrees and *k-d* binary trees as appendixes.)

[3.24] A. Unnikrishnan, P. Shankar, Y.V. Venkatesh, Threaded Linear Hierarchical Quadtrees for Computation of Geometric Properties of Binary Images, *IEEE Transactions on Software Engineering*, Vol. 14, No. 5, pp 659-665, May 1988.

[3.25] C. Anedda and L. Felician, P-Compressed Quadtrees for Image Storing, *The Computer Journal*, Vol 31, No 4, 1988

[3.26] T.C. Henderson, E. Triendl, Storing Feature Descriptions as 2-D Trees, *IEEE Transactions on Geoscience and Remote Sensing*, Vol. GE-24, No. 2, Mar 1986.

[3.27] H. Samet, Recent developments in the use of Hierarchical Data Structures for Image Databases, *AUSGRAPH 88 Proceedings*, pp 207-219, July 1988.

[3.28] A.N. Netravali and B.G. Haskell, *Digital Picture Representation and Compression*, Plenum Press, New York, pp 147-244, 1988.

[3.29] G.X. Ritter, J.N. Wilson, J.L. Davidson, Data Compression of Multispectral Images, *SPIE Applications of Digital Image Processing X*, Vol 829, pp 58-64, 1987.

[3.30] K.V. Shettigara and C.A. McGilchrist, A Principle Component and Canonical Correlation Hybrid Technique for Change Detection in Two-Image Sets, *IEEE Proceedings of the Australian Symposium on Signal Processing and Applications 89*, pp 47-52, April 1989.

[3.31] D.R. Ahlgren, J. Crosby, D.Erigat, Compression of digitised images for transmission and storage applications, *SPIE Image Processing, Analysis, Measurement, and Quality*, Vol 901, pp 105-113, 1988.

[3.32] I. Dinstein, K. Rose, A. Heiman, Variable Block-size Transform Image Coder, *9th International Conference on Pattern Recognition*, Vol. 1, pp 613-617, Nov 1988.

[3.33] R.M. Gray, Vector Quantization, *NASA - Proceedings of the Scientific Data Compression Workshop*, NASA Conference Publication 3025, pp 205-231, May 1988. (*a survey of Vector Quantisation*)

[3.34] C.N. Manikopoulis and H. Sun, Activity Index Threshold Classification in Adaptive Vector Quantization, *1988 IEEE Internal Conference on Communications*, Vol.3, pp 1235-1244, Jun 1988.

[3.35] H.-S. Wu and H.-B. Chen, A DCT Gain-Shape Vector Quantizer for Image Coding, *1988 IEEE Internal Conference on Communications*, Vol.3, pp 1245-1252, Jun 1988.

[3.36] E.A. Riskin, T. Lookabaugh, P.A. Chou, R.M. Gray, Applications of Variable Rate Vector Quantization to Speech and Image Coding, *1988 IEEE International Symposium on Information Theory*, Jun 1988.

[3.37] R. Hecht-Nielsen, Applications of Counterpropagation Networks, *Neural Networks*, Vol 1, No 2, pp 131-139, 1988

[3.38] S.P. Luttrell, Image compression using a neural network, *Proceedings of IGARSS '88 Symposium, Edinburgh, Scotland*, pp 1515-1516, Sept. 1988

[3.39] M. Barnsley, *Fractals Everywhere*, Academic Press Inc., San Diego, CA, 1988.

[3.40] M. Barnsley and A. Sloan, A Better Way to Compress Images, *Byte*, pp 215-223, Jan.1988. (the same article appears in Australian Personal Computer, pp 75-93, Feb 1988)

[3.41] G. Zorpette, Fractals: not just another pretty picture, *IEEE Spectrum*, pp 29-31, Oct. 1988.

[3.42] B.A. Cipra, Image Capture by Computer, *Science*, Vol. 243, pp 243-244, 10 Mar. 1989.

[3.43] B.D. Goel and S.C. Kwatra, A Data Compression Algorithm for Color Images Based on Run-length Coding and Fractal Geometry, *1988 IEEE Internal Conference on Communications*, Vol.3, pp 1253-1256, Jun 1988.

[3.44] D. Moran and O.J. Morris, Region and Texture coding of TV Pictures, pp 536-540.

[3.45] J.H. Lee and H.T. Liu, Digital Image Coding with High Compression Ratio, pp 595-598

[3.46] M. Todd and R. Wilson, An Anistropic Multi-Resolution Image Data Compression Algorithm, IEEE Acoustic Speech and Signal Processing, 14 th INternational Conference, pp 1969-1972, May 1989.

[3.47] P.G. Rogers, Image Compression using the Method of Convex Projections, *ISSPA 87, Signal Processing, Theories, Implementations and Applications*, pp 650-654, Aug 1987.

[3.48] M. Derkaoui, et. al , Tools for Image Compression, *Proceedings of IGARSS '88 Symposium*, Edinburgh, Scotland, pp 1515-1516, Sept. 1988.

[3.49] R.L. Baker, A VLSI chip set for real time vector quantisation of image sequences, *NASA - Proceedings of the Scientific Data Compression Workshop*, NASA Conference Publication 3025, pp 419-437, May 1988.

[3.50] S.G. Mallat, A Theory for Multiresolution Signal Decomposition: The Wavelet Representation, *IEEE Transactions on Pattern Analysis and Machine Inteilegence*, Vol 11, No 7, Jul 1989.

APPENDIX 1

Codd's 12 Relational Rules[1]

Rule 0

For any system that is advertised as, or claimed to be, a relational database management system, that system must be able to manage databases entirely through its relational capabilities

Rule 1: The Information Rule

All information in a relational database is represented explicitly at the logical level and in exactly one way - by values in tables.

Rule 2: Guaranteed Access Rule

Each and every datum (atomic value) in a relational database is guaranteed to be logically accessible by restoring to a combination of a table name, primary key value and a column name.

However in accordance with the provisions of ISO/ANSI SQL Level 2 tables are permitted which do not have a primary key constraint.

Rule 3: Systematic Treatment of Missing Information

Null values (distinct from the empty character string or a string of blank characters and distinct from zero or any other number) are supported in fully relational DBMS for representing missing information in a systematic way, independent of data type.

Rule 4: Dynamic On-line Catalog based on the Relational Model

The database description is represented at the logical level in the same way as ordinary data, so that authorized users can apply the same relational language to its interrogation as they apply to the regular data.

Rule 5: Comprehensive Data Sub-Language Rule

A relational system may support several languages and various modes of terminal use (for example , the fill-in-the-blanks mode).However there must be at least one language whose statements are expressible, per some well-defined syntax, as character strings and that is comprehensive in supporting all the following items:

- *Data definition*
- *View definition*
- *Data manipulation (interactive and by program)*
- *Integrity constraints*
- *Authorization*
- *Transaction boundaries (begin, commit and rollback)*

Rule 6: View Updating Rule

All views that are theoretically updatable are also updatable by the system.

[1] E.F.Codd, Is your DBMS really relational?, Computerworld October 14, 1985

Rule 7: High-level Insert, Update and Delete

The capability of handling a base relation[1] or a derived relation[2] as a single operand applies not only to the retrieval of data but also to the insertion, update, and deletion of data.

Rule 8: Physical Data Independence

Application programs and terminal activities remain logically unimpaired whenever any changes are made in either storage or access methods.

Rule 9: Logical Data Independence

Application programs and terminal activities remain logically unimpaired when information preserving changes are made to the base table.

Rule 10: Integrity Independence

Integrity constraints specific to a particular relational database must be definable in the relational data sub-language and storable in the catalog, not in the application programs.

Domains provide datatype and integrity constraint definitions that may be shared by many columns. Triggers can be used for referential integrity, column or table integrity. Update triggers can reference both old and new data, making it possible to define what C.J.Date calls *transition constraints.*

Rule 11: Distribution Independence

A relational DBMS has distribution independence.

This is perhaps one of the areas of fastest growth occuring in RDBMSs. Many systems now have multi-database access (joins can be performed across databases), distributed transaction processing (two phase commit protocol) and remote database access. Other distributed features which may be desirable and supported to varying extents are location transparency, distributed query processing, fragmentation transparency (individual relations may be divided by region and placed in different databases) and replication transparency (copies of relations or fragments are maintained automatically by the system in a local database to improve response time to queries).

Rule 12: Nonsubversion Rule

If a relational system has a low-level (single -record-at-a-time) language, that low level cannot be used to subvert or by pass the integrity rules and constraints expressed in the higher level relational language (multiple-records-at-a-time).

---

[1] A *base relation* is one which is defined by an SQL CREATE TABLE statement , or its equivalent.

[2] A *derived relation* is one which is defined by an SQL SELECT expression, or its equivalent (i.e. a view or a query result)

APPENDIX 2

Facsimile Encoding Schemes - The CCITT Standards

The CCITT (Consultative Committee on International Telephone and Telegraph) defines international standards for facsimile encoding an transmission. There are four groups of standards. Groups I and II are definitions for analogue devices, groups III and IV are definitions for digital devices. Since facsimile machines are basically binary flatbed scanners groups III and IV are 1-bit compression standards.

CCITT III encoding specifies two ways of compressing a bit map. The first technique is one-dimensional. Each scan line is encoded using run-length encoding and then further compressed using a Huffman code specified by the standard. This modified Huffman-encoded method achieves compression ratios of about 4:1 to 10:1. The second technique is a two dimensional method called READ (relative element address designate). It takes the one dimensional compressed image and encodes each line based on the differences from the previous line. The standard encodes and transmits in blocks of from two to seven lines and increases the compression to ratios of about 8:1 to 16:1.

CCITT IV standard builds on the Group III encoding technique. It does not transmit error correction data because this is handled by the transmission system. The Group IV facsimile machine assumes that the first line in an image is a row of white pixels. It then encodes each line as a series of changes from the previous line (using READ), and the entire image is transmitted in one large block. This results in compression ratios from 10:1 to 30:1.

DISTRIBUTION

	Copy No.
DEPARTMENT OF DEFENCE	
Defence Science and Technology Organisation	
Chief Defence Scientist	
First Assistant Secretary Science Policy (FASSP)	
Counsellor, Defence Science, London	Cnt Sht Only
Counsellor, Defence Science, Washington	Cnt Sht Only
Director, Surveillance Research Laboratory	1
Chief, (Author's Division)	1
Principal Officer, (Author's Group)	5
DSTO Salisbury, Main Library	2
Librarian, Technical Reports Centre, Defence Central Library, Campbell Park	1
OIC Document Exchange Centre, DISB (for microfiche copying then destruction)	1
Joint Intelligence Organisation (DSTI)	1
Librarian, Defence Signals Directorate Melbourne	1
Document Exchange Centre, DISB, for National Library of Australia	1
US Defense Technical Information Center (DTIC)	12
UK Defence Research Information Centre (DRIC)	2
Director Scientific Inrto Services (Canada)	1
NZ Ministry of Defence	1
Author	10

DOCUMENT CONTROL DATA SHEET

Security classification of this page : UNCLASSIFIED

1 DOCUMENT NUMBERS		**2 SECURITY CLASSIFICATION**	
AR Number :	AR-005-968	a. Complete Document :	Unclassified
Series Number :	SRL-0047-TM	b. Title in Isolation :	Unclassified
Other Numbers :		c. Summary in Isolation :	Unclassified
		3 DOWNGRADING / DELIMITING INSTRUCTIONS	

4 TITLE

STATUS REPORT ON IMAGE INFORMATIONS SYSTEMS AND IMAGE DATABASE TECHNOLOGY

5 PERSONAL AUTHOR (S)

M. Fiebig

6 DOCUMENT DATE

December 1989

7 7.1 TOTAL NUMBER OF PAGES 57

7.2 NUMBER OF REFERENCES 88

8 8.1 CORPORATE AUTHOR (S)

Surveillance Research Laboratory

8.2 DOCUMENT SERIES and NUMBER
Technical Memorandum
0047

9 REFERENCE NUMBERS

a. Task :

b. Sponsoring Agency :

10 COST CODE

11 IMPRINT (Publishing organisation)

Defence Science and Technology Organisation

12 COMPUTER PROGRAM (S)
(Title (s) and language (s))

13 RELEASE LIMITATIONS (of the document)

Approved for Public Release.

Security classification of this page : UNCLASSIFIED

Security classification of this page : UNCLASSIFIED

14 ANNOUNCEMENT LIMITATIONS (of the information on these pages)

No limitation

15 DESCRIPTORS

a. EJC Thesaurus Terms

Data Base Management Systems
Data Compression
Computer Architecture

b. Non - Thesaurus Terms

16 COSATI CODES

0062B

17 SUMMARY OR ABSTRACT
(if this is security classified, the announcement of this report will be similarly classified)

The contents of this report form part of a study to determine guidelines for the establishment of databases containing digital imagery and ancillary information. It begins with an introduction to the basic concepts of databases. This provides the background to the terminology commonly used in this report and other related literature. A review of current commercially available database systems is made, drawing principally on a market study conducted in mid 1989. Of specific interest is the extent to which image databases are being utilised and the applications which use digital imagery (or pictures) in database management systems (DBMS). Finally, there is an overview of image compression techniques, considering the current trends and future directions of the field.

Security classification of this page : UNCLASSIFIED

The official documents produced by the Laboratories of the Defence Science and Technology Organisation Salisbury are issued in one of five categories: Reports, Technical Reports, Technical Memoranda, Manuals and Specifications. The purpose of the latter two categories is self-evident, with the other three categories being used for the following purposes:

Reports	:	documents prepared for managerial purposes.
Technical Reports	:	records of scientific and technical work of a permanent value intended for other scientists and technologists working in the field.
Technical Memoranda	:	intended primarily for disseminating information within the DSTO. They are usually tentative in nature and reflect the personal views of the author.